BenchCouncil Transactions on Benchmarks, Standards and Evaluations

TBench
Volume 2, Issue 1
2022

ISSN: 2772-4859

Contents

Contents lists available at ScienceDirect

BenchCouncil Transactions on Benchmarks, Standards and Evaluations

journal homepage: https://www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/

Three laws of technology rise or fall

Jianfeng Zhan

Research Center for Advanced Computer Systems, Institute of Computing Technology, Chinese Academy of Sciences, China

ARTICLE INFO

Keywords:
Technology laws and corollaries
Law of technology inertia
Law of technology change force
Law of technology action and reaction
Corollary of measurement of technology change force
Corollary of technology breakthrough
Corollary of technology openness
Corollary of technology monopoly
Corollary of technology business opportunity
RISC-V

ABSTRACT

Newton's laws of motion perfectly explain or approximate physical phenomena in our everyday life. Are there any laws that explain or approximate technology's rise or fall? After reviewing thirteen information technologies that succeeded, this article concludes three laws of technology and derives five corollaries to explain or approximate the rise or fall of technology. Three laws are the laws of technology inertia, technology change force, and technology action and reaction. Five corollaries are the corollaries of measurement of technology change force, technology breakthrough, technology monopoly, technology openness, and technology business opportunity. I present how to use the laws and the corollaries to analyze an emerging technology—the open-source RISC-V processor. Also, I elaborate on benchmarks' role in applying those laws.

1. Introduction

Basic research, including both basic scientific research and basic technological research [1], generates scientific and technical knowledge. At the same time, technology is the application of scientific and technical knowledge for practical purposes, especially in the industry [2]. Researchers published many scientific and technical papers or proposed massive new concepts or ideas. Upon scientific and technical knowledge, engineers develop massive technologies, e.g., tools, platforms, products, and services in the information technology (IT) industry. But few technologies matured and succeeded! Why? Are there any laws that can describe or approximate technology rise or fall as Newton's laws of motion perfectly explain or approximate physical phenomena in our everyday life?

The National Research Council of the US [3–5] published a series of white papers on the patterns of innovation in IT, evidenced by many case studies. The white papers concluded IT has a long, unpredictable incubation period, and the university and industry have a complex partnership in innovations. Meanwhile, they observed the by-product, resurgence, and confluence phenomena in the IT innovations [3–5]. Also, many scholars pondered the nature of technology. Jacques Ellul [6] and Langdon Winner [7] concluded the philosophical doctrine of technological determinism [8]. Lynn Townsend White declined to accept this technological omnipotence [8]. Instead, he claimed that a technical device "merely opens a door, it does not compel one to enter". [8,9]. Melvin Kranzberg [8] concluded a series of observations deriving from a longtime study, Kranzberg's Six Laws of Technology. These observations, quests, or arguments help understand the development of technology and its interactions with sociocultural change [3–5,8]. Still, they fail to provide a qualitative or quantitative approach to explain or approximate the rise or fall of technology.

After analyzing thirteen successful IT, I conclude three laws analogous to Newton's laws of motion to explain technology's rise and fall in Section 2. The first law is on the obstacle to new technology: technology inertia. Not only end-users but also industry users stick to the existing technology. The user size will keep constant unless a non-zero net technology change force acts on it. The second law reveals where the power of new technology comes from. The change of user size is proportional to the net technology change force. The corollary of measurement of technology change force is how to measure the net change force. Only by creating a brand-new technology or improving an existing technology in terms of user experience, costs, efficiency, or other fundamental dimensions by several orders of magnitude can the new technology generate a positive change force. The third law explains how existing and emerging technologies compete. When the net technology change force is positive, the emerging technology rises and the existing technology falls, or else the existing technology keeps. In Section 3, I derive two corollaries, named the corollaries of technology breakthrough and technology monopoly, to explain how a new technology breakthroughs and gains monopoly, respectively. In addition, two corollaries called the corollaries of technology openness, and technology business opportunity explain why an open technology gains edge over a closed one and how a technology achieves business opportunities, respectively. Table 1 summarizes the three laws and five

E-mail address: zhanjianfeng@ict.ac.cn.
URL: http://www.benchcouncil.org/zjf.html.

https://doi.org/10.1016/j.tbench.2022.100034

Available online 15 March 2022

Table 1
The summary of laws of technology.

Law or corollary name	Formula
Law of technology inertia	$\Delta U_t = U_{(t+\Delta t)} - U_t$ $\Delta U_t = 0$, U *is a natural number*
Law of technology change force	$\Delta U_t \propto F_t$, U *is a natural number*
Law of technology change action and reaction	$F_{Emerging} = -F_{Existing}$
Corollary of measurement of technology change force	$F_t = F_{Create} + F_{Learn} + F_{Ecosystem}$ or $F_t = F_{Experience} + F_{Cost} + F_{Efficiency} + F_{Other} + F_{Learn} + F_{Ecosystem}$
Corollary of technology breakthrough	$B = F/U$, U *is a natural number*
Corollary of technology openness	$P = p * U_i,\ c = C/U_i$ $P = p * U_i/M,\ c = C * M/U_i$ $P = p * U_i/N,\ c = C * N/U_i$
Corollary of technology monopoly	$U = \Sigma\Delta U_t \propto F$
Corollary of technology business Opportunity	/

Table 2
The explanations of symbols in Table 1.

Symbol	Explanation
Δ	Difference operator
$\propto$	Proportional operator
Σ	Summation operator
U_t	User size varying with time
U_i	Size of industry users
F_t	Net technology change force varying with time
$F_{Emerging}$	Change force acting on emerging technology
$F_{Existing}$	Change force acting on existing technology
F_{Create}	Change force resulted from creating a brand-new technology
F_{Learn}	Change force resulted from learning cost
$F_{Ecosystem}$	Change force resulted from ecosystem deviation
$F_{Experience}$	Change force resulted from user experience
F_{Cost}	Change force resulted from cost
$F_{Efficiency}$	Change force resulted from efficiency
F_{Other}	Change force resulted from other fundamental dimensions
B	Technology breakthrough
P	Gross productivity
C	Total cost
M	Number of decoupled supply chains
N	Number of nations
c	Cost of each contributor (industry user)
p	Productivity of each contributor (industry user)

corollaries together. Table 2 explains the symbols in the formula in Table 1.

I want to emphasize that these laws differ from Newton's laws. The former is difficult to verify through quantitative experiments under repeatable or reproducible settings because its settings are highly complex. Some laws stand the shoulders of giants, quoted from Newton's saying. For example, the previous work [10] presented observations similar to the corollaries of measurement of technology change force and technology monopoly. I point out the difference in each subsequent subsection. My contribution is to propose a simple but powerful theory framework (three laws and five corollaries) to explain the technology's rise or fall.

Section 4 uses the three laws and the four corollaries to predict an emerging technology RISC. Section 5 discusses the role benchmarks played in applying those laws and corollaries. Section 6 presents the related work. Finally, I conclude Section 7.

2. The three laws of technology

After analyzing the technology in Table 3, this section presents the laws of technology inertia, technology change force, and technology action and reaction, respectively.

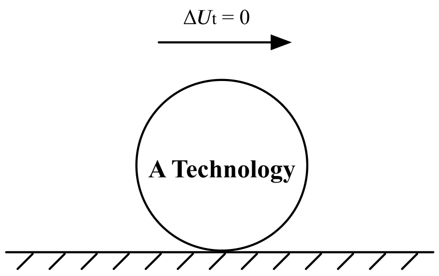


Fig. 1. The law of technology inertia.

2.1. *The law of technology inertia (the first law of technology)*

Newton's first law states that an object moves with a velocity that is constant in magnitude and direction unless a non-zero net force acts on it [11]. Similar to Newton's first law, I presume the existence of the technology inertia – the user size U_t of a technology will keep constant unless a non-zero net technology change force F_t acts on it. Out of learning pressure and usage habits, end-users U_e stick to existing products, tools, platforms, and services called consumer inertia. Industrial users U_i adhere to existing products, tools, platforms, and services for investment protection, called ecosystem inertia. The consumer inertia and ecosystem inertia together constitute the technology inertia. The user size in the first law is a normalized number referring to both end-users U_e and industrial users U_i. Eq. (1) quantitatively states this law. Fig. 1 presents the law of technology inertia. The first law is a hypothesis for the general trend. Someone may argue the exception case also holds. I do not doubt it. But the law of technology inertia is a hypothesis at a macroscopic level, not a microscopic level.

The law of technology inertia: the user size U_t will keep constant unless a non-zero net technology change force F_t acts on it.

$$\begin{aligned} &\Delta U_t = U_{(t+\Delta t)} - U_t \\ &\Delta U_t = 0\ ,\ U \textit{ is a natural number} \end{aligned} \tag{1}$$

A technology's ecosystem is complex, needing an exhausting analysis. I will elaborate on them in Section 4.

2.2. *The law of technology change force (the second law of technology)*

Newton's second law states that the acceleration of an object is directly proportional to the net force acting on it and inversely proportional to its mass [11]. Like Newton's second law, I presume the change of the user size is proportional to the net technology change force, which I call the law of technology change force. Eq. (2) states this relationship qualitatively. The user size increases when the net change force is positive, while the user size decreases when the net change

Table 3
The analysis of thirteen successful IT.

Technology	Rivals	F_{Learn}	$F_{Ecosystem}$	F_{Create}	F_{Cost}	$F_{Efficiency}$	$F_{Experience}$	F_{Other}	Closed/Open	Supply chain
Deep learning	Shallow neutral networks	0	<0	/	<0	/	/	>0 (accuracy)	Open	Coupling
WWW	No	<0	<0	>0	/	/	/	/	Open	Coupling
Google	No	<0	<0	>0	/	/	/	/	Closed	Decoupling
Facebook	No	<0	<0	>0	/	/	/	/	Closed	Decoupling
Internet	No	<0	<0	>0	/	/	/	/	Open	Coupling
RAID	Single large expensive disk	0	<0	/	>0	>0	/	/	Open	Coupling
Android	Windows Mobile, Symbian, iOS, Linux	<0	0 (with respect to Linux)	/	>0	/	>0	>0 (Google ecosystem)	Open	Coupling
iOS	Windows Mobile, Symbian	<0	<0	/	<0	/	>0	>0 (AppStore)	Closed	Coupling
Windows	DOS	<0	0	/	/	/	>0	/	Closed	Coupling
Linux	UNIX	0	0	/	>0	/	/	/	Open	Coupling
UNIX	Multics	<0	<0	/	/	>0	/	>0 (standard)	Closed	Coupling
ARM	X86, RISC	0	0 (with respect to RISC)	/	/	>0	/	>0 (energy efficiency)	Closed	Decoupling
RISC	CISC	0	0	/	/	>0	/	/	Closed	Decoupling

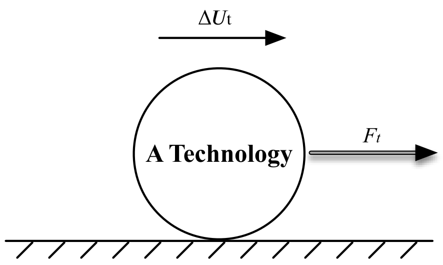


Fig. 2. The law of technology change force.

force is negative. When the net change force is zero, the user size keeps constant according to Eq. (1). Please note that in Eq. (2), U is a natural number. With a negative change force acting, when U decreases to zero, it will trigger a net change force $F_t = 0$ because a negative user size has no physical meaning. I can not quantify the relationship between ΔU and F_t accurately as it is much more challenging. Fig. 2 presents the law of technology change force.

The law of technology change force: The change of user size ΔU_t is proportional to the net technology change force F_t.

$$\Delta U_t \propto F_t\ ,\ U \text{ is a natural number} \tag{2}$$

Creating a brand-new technology will generate a positive change force F_{create}. However, from proposing a new concept to falling into the ground, it has a long, unpredictable incubation period separating into multiple phases of innovation: research exploration, initial commercial deployment, and eventual business breakthrough [3,5]. My law of technology change force gives another explanation. Though creating a brand-new technology will generate a positive change force, the learning cost (consumer inertia) of end-users F_{Learn} and ecosystem inertia of industry users $F_{Ecosystem}$ will generate negative change forces. As these different components counter each other, the net change force will vary and finally decide its rise or fall. The steam engine, plane, and computer industries all witnessed this process. Table 3 presents IT examples like WWW, Internet, Google, Facebook.

Melvin Kranzberg [8] explains the nature of creating a brand-new technology in Kranzberg's Second Law: Invention is the mother of necessity. He took the automobile as an example of how a successful technology requires auxiliary technologies to make it fully effective [8]: the automobile brought whole new industries by their need for rubber tires, petroleum products, and new tools and materials; Large-scale commercial automobile deployment demanded roads, highways, garages, parking lots, traffic signals, and parking meters [8].

For an existing technology, only improving its user experience, cost, efficiency by several orders of magnitude can generate a positive change force $F_{Improve}$ that can break through the technology inertia, or else it will generate a negative change force. Thiel et al. proposed a rule of thumb in [10]: Proprietary technology must be at least ten times better than its closest substitute in some important dimension to lead to real monopolistic advantage. Otherwise, it will probably be perceived as a marginal improvement. Table 3 presents IT examples like deep learning, RAID, etc.

In improving an existing technology, a new or different ecosystem will generate a negative change force $F_{Ecosystem}$, which is the side effect of ecosystem inertia. Meanwhile, different use which results in a learning cost will generate a negative change force F_{Learn}. When two rivals can substitute without causing any overhead for end-users and industrial users, the net change force is zero, and the earlier technology gets a head start. I conclude the above discussion as a corollary, named the law of measurement of technology change force. Eq. (3) states the net change force F_t is the sum of F_{Create}, F_{Learn} and $F_{Ecosystem}$ or the sum of $F_{Improve}$, F_{Learn} and $F_{Ecosystem}$.

Corollary of Measurement of Technology Change Force: Only by creating a brand-new technology (F_{Create}) or improving existing technology ($F_{Improve}$) in terms of user experience ($F_{Experience}$), cost (F_{Cost}), efficiency ($F_{Efficiency}$), or other fundamental dimensions (F_{Other}) by several orders of magnitude can it generate a positive net change force, or else it will generate a negative change force. A new ecosystem or the deviation between emerging and existing technology ecosystems will generate a negative change force $F_{Ecosystem}$. Different use, which results in an end-user learning cost, will generate a negative change force F_{Learn}. The net technology change force is zero when two rivals can substitute without causing any overhead for end-users and industrial users.

$$\begin{aligned} &F_t = F_{Create} + F_{Learn} + F_{Ecosystem} \\ &or \\ &F_t = F_{Experience} + F_{Cost} + F_{Efficiency} + F_{Other} + F_{Learn} \\ &+ F_{Ecosystem} \end{aligned} \tag{3}$$

2.3. The law of technology action and reaction (the third law of technology)

Newton's third law states that if object one and object two interact, the force F_{12} exerted by object one on object two is equal in magnitude but opposite in direction to the force F_{21} exerted by object 2 on object 1 [11]. Similar to Newton's third law, I state the third law of technology: technology change forces acting on the emerging technology ($F_{Emerging}$) and reacting on the existing technology ($F_{Existing}$) equal in magnitude but opposite in direction. Fig. 3 presents the law of technology action and reaction. A change force acts on the emerging technology while a change force in the same magnitude but different directions reacts to the existing technology. When the change force acting on the emerging technology is positive, the emerging technology rises (user size increases), and the existing technology falls (user size declines). Eq. (4) states this relationship quantitatively.

The law of technology action and reaction: Under the normal conditions, technology change forces acting on the emerging technology ($F_{Emerging}$) and reacting on the existing technology ($F_{Existing}$) are

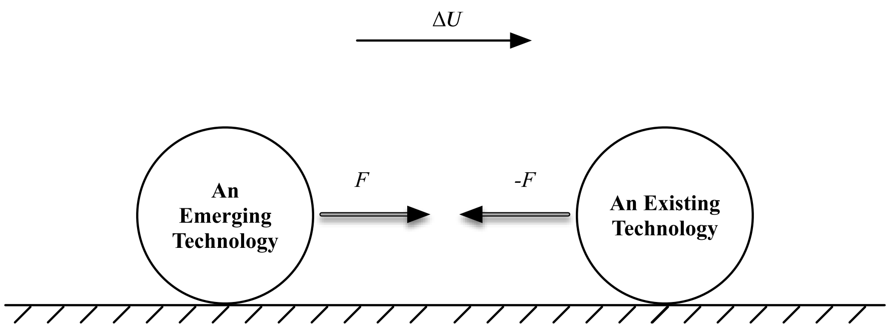


Fig. 3. The law of technology action and reaction. In this figure, the technology change force is positive.

equal in magnitude but opposite in direction.

$$F_{Emerging} = -F_{Existing} \tag{4}$$

Furthermore, I clarify the normal conditions where the third law of technology holds. First, it should exclude decoupling of the supply chain, which the pandemic accelerates. The decoupling of the supply chain is why the same technology is replicated, or superior and inferior technology coexist in different nations. Second, it should exclude the case where the government has total control over the production and pricing of goods and services. In the latter case, the market will weaken the technology change forces. Third, in a normal condition, it must warrant the protection of intellectual properties and the punishment of business fraud. Superior technology can be mirrored or stolen without paying prices without protecting intellectual properties. Meanwhile, tolerating business fraud weakens the signals emitted by technology change forces.

I take two computer examples in the shift from server to desktop to mobile to explain it. The creation of standardized, vendor-independent operating systems, such as UNIX and its open-source clone, Linux, shields the new architecture's change, making the ecosystem inertia against new architecture disappear. And hence it lowered the cost and risk of bringing out a new architecture [12]. According to Corollary One, $F_{Ecosystem} = 0$.

The RISC-based computers raised the performance bar, and its efficiency generate a positive change force ($F_{Efficiency} > 0$) that broke through the technology inertia of the prior architectures like Digital Equipment VAX, forcing the latter to disappear [12] (RISC vs. VAX). The late 1990s witnessed the soaring transistor counts, which provided a survival chance for Intel X86 architecture. As the hardware overhead becomes negligible, the latter translates 80 × 86 instructions into RISC-like ones internally to keep up with the RISC ecology – leveraging many of the innovations first pioneered in the RISC designs and pacing with the RISC performance ($F_{Ecosystem} = 0$ and $F_{Efficiency} = 0$). In this context, the RISC architecture generates an almost zero net change force against the X86 architecture (RISC vs. X86) ($F = 0$). X86 survive. After the emergence of mobile applications, the constraints of power and silicon area make the x86-translation overhead unacceptable and lead to the dominance of a RISC architecture, ARM [12]. ARM's reducing cost ($F_{Cost} > 0$) and improving efficiency ($F_{Efficiency} > 0$) while compatible with the ecosystem ($F_{Ecosystem} = 0$) generate a positive change force that can break through the technology inertia of x86 (ARM vs. x86).

Microsoft and other companies predicted the shift from desktop to mobile. The competition between Windows Mobile and Android is governed by the law of technology change force. The Android OS bases Linux, which worldwide world-class engineers contribute, offering the open-source operating system to third-party mobile manufacturers for free [13]. In contrast, Windows Mobile chose the opposite direction: closed-source Windows as the basis and non-free use (Android is free, while Windows Mobile costs manufacturers \$15 to \$25 a phone in 2009 when Windows Mobile dominated Android [14]). Meanwhile, Android is designed for mobile: Android's software is intended for modern screens you tap with a finger, while Windows Mobile was built for use with a stylus [14]. Both cost ($F_{Cost} > 0$) and user experience ($F_{Experience} > 0$) edges of Android shook the technology inertia, while Windows Mobile faded (Android vs. Windows Mobile).

Social networking examples also confirm this law. Social networking applications like Facebook, Twitter, WeChat fell into the ground, bringing devastating advantages and benefits. Many companies want to copy the success. For example, several E-commerce service providers and mobile phone giants with massive user bases, strong financial support, and world-class engineers are devoted to building social-networking applications. Predictably, they all failed. Out of learning pressure and usage habits, users stick to Facebook, Twitter, or WeChat. Moreover, Facebook, Twitter, or WeChat open their services and API to third parties to build the ecosystem. The third-parties service providers also are accustomed to adhering to Facebook, Twitter, or WeChat for investment protection. Why are there similar social networking applications in the US and China? I discuss this case above.

3. Four corollaries of technology

I discuss several corollaries as follows.

3.1. Corollary of technology breakthrough

Corollary of technology breakthrough: The technology breakthrough B is defined as the net change force F divided by the target or real user size U. There are three ways to raise technology breakthrough: increase the net change force, satisfy a smaller target users' requirement through focusing on a more minimal set of functions, ensure the emerging technology's ecosystem is compatible with the existing one.

$$B = F/U \text{ , } U \textit{ is a natural number} \tag{5}$$

I use a physics concept, Pressure P, to explain this law. If F is the magnitude of a force exerted perpendicular to a given surface of area A, then the average Pressure P is the force divided by the area [11]. When a new technology emerges, it needs to overcome the technology inertia and contact end-users U_e and industry users U_i. Here, I use the user size U to measure the given surface of the contact area. So, according to the pressure definition, the breakthrough B is defined in Eq. (5). The increase of the net change forces will raise the breakthrough. Ensuring the emerging technology compatible with the ecosystem of the existing technology will ensure $F_{Ecosystem} = 0$ according to the corollary of measurement of technology change force.

Focusing on a minimum set of functions compared to a full-fledged one has several benefits. First, a smaller contact area A (smaller target users U) with the same net change force will exert a larger pressure (breakthrough B). Second, it is much easier to improve user experience, cost, or efficiency (raise the net change force). Last but not least, it will control the cost, which is also paramount for innovation. Of course, a minimum set of functions is the lower bound of granularity, or else too simple technology cannot satisfy the users' basic requirements.

I still take the mobile operating system as an example. When iOS and Android emerged, Symbian and Windows Mobile dominated the mobile operating system market. In January 2007, Steve Jobs announced the original iPhone, a touch-based smartphone without a physical keyboard, controlled with a finger without needing a stylus [15]. The first version of iOS (no official name) was quite limited, focusing on the fresh user experience of introducing a complete touch screen. Contrasting, most users are accustomed to typing on a hardware keyboard at that time. The first iOS version lacked many features those alternative systems already had [16].

While the first version of the iPhone featured a closed ecosystem with the strength of clean and simple design, the first Android phone built the edges of both cost and user experience, brought a free and open-source ecosystem, personality, and the ability to customize user experience [17]. The first version of Android features several unique characteristics [17] (1) choose open-source Linux as a basis, and it is free; (2) allow both the third parties and users to customize the systems, and even additional applications and features; (3) launch Android Market (Google Play) to build application ecosystem; (4) integrate the Google's already robust ecosystem, constantly expanding software. Android and iOS built full-fledged functions later. Those two typical examples witnessed the law of technology breakthrough.

3.2. *Corollary of technology monopoly*

Corollary of technology monopoly: The greater the net technology change force F, the higher the technology monopoly.

According to the second law, the change of user size ΔU is proportional to the net technology change F. For the same time interval T, the greater the net technology change force, the higher user size $U = U_t + \Delta U$. Eq. (6) states this relationship. Naturally, much superior technology gains a higher technology monopoly. Please note that in Eq. (6), F is a normalized value as it may vary in different intervals. In [10], monopoly means the kind of company that is so good at what it does that no other firm can offer a close substitute. Currently, Android and iOS gain the technology monopoly among the mobile operating systems.

$$U = \Sigma \Delta U_t \propto F \tag{6}$$

3.3. *Corollary of technology openness*

Corollary of technology openness: Contrasted with a closed ecosystem controlled by one entity, allowing the division of labor among contributors who share an open technology ecosystem improves the gross productivity P and lowers the cost c amortized on each contributor (U_i). Decoupling the supply chain (M) and export control of technology between nations(N) will increase the cost and lower productivity. Standards will decrease the cost of collaboration in industry contributors.

$$P = p * U_i \qquad c = C/U_i \tag{7}$$

$$P = p * U_i/M \qquad c = C * M/U_i \tag{8}$$

$$P = p * U_i/N \qquad c = C * N/U_i \tag{9}$$

There are N nations, M decoupled supply chains, U_i technology contributors. N, M, U_i are natural numbers. For a specific technology, the gross productivity is P, and the total cost is C. I presume there is the same size of the technology contributors in each nation or decoupled supply chain. The productivity of each contributor is the same, $p\ jobs\ per\ day$. As the goal is to perform qualitative analysis, this presumption is reasonable. Eq. (7) states the total productivity P and the cost of each contributor (in total, U_i contributors). Eq. (8) states the gross productivity P and the cost of each contributor within each decoupled supply chain (in total, M decoupled supply chains). Eq. (9) states the total productivity P and the cost of each contributor in the nation (in total, N nations) subject to export control.

The explanation is also intuitive. Allowing the division of labor among contributors, each contributor performs the job in parallel, hence lowering the cost on average. Meanwhile, the gross productivity will improve despite the collaboration overhead. Eq. (7) shows that the gross productivity P is proportional to the number of contributors U_i, and the cost of each contributor c is inversely proportional to the number of contributors U_i. That is why many industries open their services and API to third parties to build the ecosystem.

Decoupling the supply chain (M) will increase the cost and lower productivity because several similar technologies have to been replicated. Eq. (8) shows that the gross productivity P of each decoupled supply chain is proportional to the number of contributors U_i and inversely proportional to the number of decoupled supply chain M; the cost of each contributor c is inversely proportional to the number of contributors U_i and proportional to the number of decoupled supply chain M.

Putting a nation subject to export control (N) will increase the cost and lower productivity of that nation because the similar technology have to been developed by one nation independently. Eq. (9) shows that the gross productivity P of the nation under export control is proportional to the number of contributors U_i and inversely proportional to the number of nations N; the cost of each contributor c within the nation under export control is inversely proportional to the number of contributors U_i and proportional to the number of nations N.

Standards will decrease the cost of collaboration in industry contributors. The standardization of interfaces, modularity, and interoperability accelerates the technology impact, enables the absorption of innovations into diverse sectors, and propels industry investment by creating network effects [5]. Metcalfe's law [18,19] can be used to quantify the network effects. The law states that the value V of a network is proportional to the square of the size n of the network [19]. A classic example of the power of well-designed interfaces is the TCP/IP suite of Internet protocols [5].

3.4. *Corollary of technology business opportunity*

Corollary of technology business opportunity: There are four technology business opportunities — (1) create a brand-new technology; (2) achieve the superior edge of cost, efficiency, user experience, or other fundamental dimensions; (3) seek a superior position in the division labor sharing a technology ecosystem; (4) opportunities arising from supply chain decoupling or export control.

This corollary summarizes the points I discussed above together. I will not repeat all of them. I only elaborate on the third opportunity. The third law of Kranzberg's Six Laws of Technology [8] reads as follows: technology comes in packages, big and small. It has been extended even further by Thomas P. Hughes. Hughes more precisely and accurately uses systems instead of packages, which he defined systems as coherent structures composed of interacting, interconnected components [20]. In seeking a superior position in the division of labor sharing a technology ecosystem, the goal is to develop a superior component that gains the edge of cost, efficiency, user experience, or other fundamental dimensions against other rival components.

4. Analyzing RISC-V

RISC-V [21] is an open standard instruction set architecture (ISA) that is based on established reduced instruction set computer (RISC) principles, which is provided under open source licenses, unlike most other ISA designs. In this section, I perform a simple analysis of the edge of RISC-V processors against other rivals in different domains (see Table 4).

Table 4
The analysis of the RISC-V processor. As Fig. 5 shows, RISC-V ISA is only a part of the RISC-V processor. The open-source license of RISC-V ISA only ensures it is partially open and coupling. A completely open and coupling RISC-V processor needs the open-source implementation of SoC, shown in Fig. 5, without violating the intellectual proprieties.

Technology	Domain	Rivals	F_{Learn}	$F_{Ecosystem}$	F_{Create}	F_{Cost}	$F_{Efficiency}$	$F_{Experience}$	F_{Other}	Closed/Open	Supply chain
RISC-V processor	Desktop	X86	0	Uncertain	/	>0	Uncertain	Uncertain	Uncertain	Partially open	Partially coupling
	Server	X86	0	Uncertain	/	>0	Uncertain	Uncertain	Uncertain	Partially open	Partially coupling
	Smart phone	ARM	0	Uncertain	/	>0	Uncertain	Uncertain	Uncertain	Partially open	Partially coupling
	IoT	No	0	Uncertain	/	>0	Opportunity	Opportunity	Opportunity	Partially open	Partially coupling

Fig. 5 presents the ecosystem of an X86, ARM, or RISC-V ecosystem. An entire ecosystem of X86, ARM, or RISC-V consists of SoC (a system on a chip), ISA (Instruction Set Architecture), OS (operating system), toolchain, middleware, and applications. SoC is an IC that integrates multiple system components onto a single chip. It often includes CPU cores, memory, input/output ports, and secondary storage, alongside other components such as a graphics processing unit (GPU) [22]. The Instruction Set Architecture (ISA) defines the interface between software and hardware. The software applications' code is converted to machine instructions, executed at the hardware level. The OS is the primary software managing the hardware resource of a computer. The toolchain is a set of programming tools to develop software, consisting of a compiler, a linker, libraries, and a debugger. Middleware provides services to applications beyond the OS, such as the database system. The application offers the service for the user.

In the desktop, server, and smartphone domains, rivals like x86 and ARM are mature. Suppose the RISC-V ecology can provide full-fledged functions similar to X86 or ARM, according to Law of Measurement of Technology Change Force, $F_{Ecosystem} = 0$. Unfortunately, it is not a trivial job. According to the law of measurement of technology change force, there are four other components of technology change force: F_{Cost}, $F_{efficiency}$, $F_{Experience}$, F_{Other}. F_{Cost} is larger than zero. But the other components are uncertain, and they may be negative. In that case, the net change force may be negative. Even the net change force is positive, in coping with this daunting challenge, the law of technology breakthrough reminds us that it is helpful to propose a minimum set of functions to generate a big breakthrough B, overcoming the inertia barrier.

The IoT domain is emerging, and the current solutions are fragmented. RISC-V processor gains a better position as there are no cemented rivals. It is much possible that RISC-V gains edges of efficiency, experience and other fundamental dimensions. The RISC-V processor has a promising future as it is open and free.

The other advantage of RISC-V is they are partially open and coupling while the other rivals are closed or decoupling. According to the law of technology openness, the division of labor among contributors significantly improves gross productivity and lowers the cost amortized on each contributor. Suppose RISC-V can have a positive net change force (see the above analysis), attracting more contributors (U_i) joining. The gross productivity P and the cost of each contributor c will increase and decrease with U_i, respectively. The other pro of RISC-V is that it will face supply chain decoupling and export control to a lesser extent (the SoC may be subject to decoupling supply chain and export control). As the law of technology openness put, decoupling supply chain and export control will increase the cost and lower productivity.

In a word, according to the corollary of technology business opportunities: there are two primary opportunities for RISC-V processors. (1) achieve the superior edges of cost, efficiency, user experience, or other fundamental dimensions; (2) opportunities arising from supply chain decoupling or export control.

5. The role of benchmark played

In IT, there are two categories of benchmarks that can play a role in evaluating technology rise or fall [23]: the first category of the benchmark – a measurement standard and the second one – the representative workloads that run on the systems under measurement.

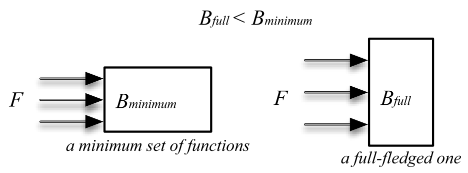


Fig. 4. The corollary of technology breakthrough.

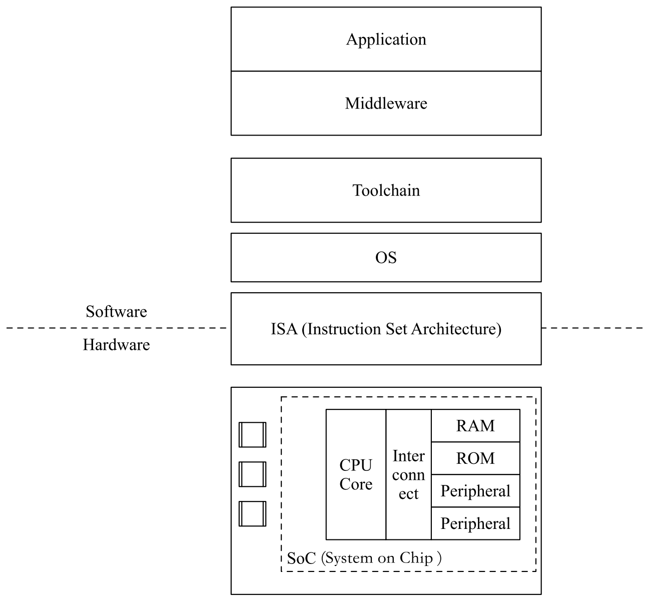


Fig. 5. The ecosystems of X86, ARM or RISC-V processors.

In the other technology domain, a similar methodology like the fifth category of benchmark – benchmarking can be developed [23]. In this context, benchmarking is the continuous process of searching the industry best practices that lead to superior performance and measuring products, services, and processes against them [23,24].

When applying the second technology law, it is best to propose a benchmark to compare against each technology's cost, efficiency, user experiences, and other fundamental dimensions, which is objective.

6. Related work

The National Research Council of the US [3–5] published a series of white papers on the innovation in IT. The first observation is that IT has a long, unpredictable incubation period separating into multiple phases of innovation: research exploration, initial commercial deployment, and eventual business breakthrough [3,5]. A striking example is the field of number theory: a branch of pure mathematics without applications for hundreds of years now became the basis for the public-key cryptography that underlies information security [3].

The second observation is that the university and industry have a complex partnership in innovations [3–5]. One pattern is that the initial ideas came from the industry, not commercialized until the university launched the research projects. For example, IBM pioneered both the concept of reduced-instruction-set computing (RISC) processors and relational databases (its System R project). The former was not commercialized until the University of California at Berkeley and Stanford University performed its Very Large-Scale Integrated Circuit (VLSI) program of the late 1970s and early 1980s [3–5]. The latter was not commercialized until the University of California at Berkeley brought this technology to where several start-up companies commercialized it [3–5]. The other pattern is that the initial ideas came from the university community, followed by industry research. The time-sharing system, which aims at making it possible to share expensive computing resources among multiple simultaneous interactive users, follows this pattern [3–5].

The third is the by-product phenomena, where collateral results, often unanticipated, are the by-product of the anticipated ones but as important as the anticipated results of research [3,5]. For example, electronic mail and instant messaging were by-products of the time-sharing systems.

The fourth is about the resurgence phenomena. Researchers' interests can fall off in areas where progress has slowed, followed by a resurgence when new ideas or enablers emerge [5]. It is difficult to predict a priori which research will pay off rapidly and take time, which typical resurgence examples like Machine learning, Formal methods, Virtual machines (VMs), and Virtual reality (VR) witnessed [5].

The fifth is about the confluence phenomena where multiple threads of IT-enabled innovation coming together have a transformative impact on a significant industry sector through converging contributions from various areas of IT innovation [5]. The impact of the Internet is the underlying canonical component that contributes to many confluence successes [5]. Confluence relies on several factors [5]: the ability to combine deep expertise about the domain in which IT is being applied with deep knowledge in IT, design and production knowledge that combines knowledge of the application and IT, and the development of new business models that take advantage of the capabilities afforded by IT.

Many scholars pondered the nature of technology. Jacques Ellul [6] and Langdon Winner [7] conclude the philosophical doctrine of technological determinism [8]: technology is pursued for its own sake and without regard to human need. Melvin Kranzberg [8] interpreted this proposition as "technology has become autonomous and has outrun human control; in a startling reversal, the machines have become the masters of man". However, some scholars declined to accept this technological omnipotence [8]. Lynn Townsend White claimed that a technical device "merely opens a door, it does not compel one to enter". [8,9] . Melvin Kranzberg [8] concluded a series of observations deriving from a longtime study, Kranzberg's Six Laws of Technology. Kranzberg's First Law reads as follows: technology is neither good nor bad; nor is it neutral, which can be interpreted in that the same technology can have quite different results when introduced into different contexts or under different circumstances [8].

A lots of previous work studies the network effect. A network effect is the effect of a network's value V is dependent on its size n (the number of its nodes) [25]. Four laws have been proposed to provide more precise definitions and characterizations of network effect [25]. They are Sarnoff's law [26], Odlyzko's law [27], Metcalfe's law [18] and Reed's law [28]. Metcalfe himself [19] used Facebook's data over the past 10 years to show a good fit for Metcalfe's law. Zhang et al. [25] expanded Metcalfe's results by utilizing the actual data of Tencent and Facebook and validated Metcalfe's Law.

7. The conclusion

This article concluded three laws of technology: laws of technology inertia, technology change force, and technology action and reaction, and derived five corollaries: corollaries of measurement of technology change force, technology breakthrough, technology monopoly, technology openness, and technology business opportunities. These laws and corollaries provide a theory framework to explain or approximate technology rise or fall. I presented how to use this theoretical framework to analyze an emerging technology – RISC-V. Also, I elaborated on benchmarks' role in applying those laws.

Acknowledgments

I am very grateful to many persons' contributions, especially Dr. Lei Wang for discussing three laws, contributing and discussing Table 3, contributing Fig. 5, and proofreading the article, Mr. Shaopeng Dai for editing this article, drawing Figs. 1–4 and compiling the references, Dr. Chunjie Luo and Dr. Wanling Gao for contributing and discussing Table 3 and proofreading this article.

References

[1] National Research Council, Funding a Revolution: Government Support for Computing Research, The National Academies Press, 1999.
[2] Oxford University Press, OxfordLanguages, https://languages.oup.com/google-dictionary-en/.
[3] National Research Council, Innovation in Information Technology, The National Academies Press, 2003.
[4] National Research Council, et al., Continuing Innovation in Information Technology, The National Academies Press, 2012.
[5] National Research Council, Information Technology Innovation: Resurgence, Confluence, and Continuing Impact, The National Academies Press, 2020.
[6] Jacques Ellul, The technological society, 1964, pp. 229–318, New York.
[7] Langdon Winner, Autonomous Technology: Technics-Out-of-Control As a Theme in Political Thought, Mit Press, 1978.
[8] Melvin Kranzberg, Technology and history:" Kranzberg's laws", Technol. Cul. 27 (3) (1986) 544–560.
[9] Lynn Townsend White, Lynn White (Jr.), Medieval Technology and Social Change, Vol. 163, Galaxy Books, 1964.
[10] Peter A. Thiel, Blake Masters, Zero to One: Notes on Startups, Or how to Build the Future, Crown Business, 2014.
[11] Raymond A. Serway, Chris Vuille, College Physics, Cengage Learning, 2011.
[12] John L. Hennessy, David A. Patterson, Computer Architecture: A Quantitative Approach, Elsevier, 2011.
[13] John Callaham, The history of android: The evolution of the biggest mobile OS in the world, 2021, https://www.androidauthority.com/history-android-os-name-789433/.
[14] Saul Hansell, Big cellphone makers shifting to android system, N.Y. Times (2009) B4.
[15] L.E.M. Staff, Original iPhone, 2007, https://lowendmac.com/2007/original-iphone/.
[16] Simon Royal, Iphone OS 1: The beginning of an era, 2017, https://lowendmac.com/2007/original-iphone/.
[17] Jessica Dolcourt, Kent German, With Google's first android phone, the iphone finally got a rival, 2018, https://www.cnet.com/tech/mobile/with-googles-first-android-phone-the-iphone-finally-got-a-rival/.
[18] George Gilder, Metcalf's law and legacy, Forbes ASAP (1993).
[19] Bob Metcalfe, Metcalfe's law after 40 years of ethernet, Computer 46 (12) (2013) 26–31.
[20] Thomas P. Hughes, Networks of Power: Electrification in Western Society, Baltimore, 1983, 1983.
[21] Andrew Waterman, Yunsup Lee, David A. Patterson, Krste Asanovi, The risc-v instruction set manual, Technical Report, California Univ Berkeley Dept of Electrical Engineering and Computer Sciences, 2014.
[22] Michael J. Flynn, Wayne Luk, Computer System Design: System-on-Chip, John Wiley & Sons, 2011.
[23] Jianfeng Zhan, Call for establishing benchmark science and engineering, BenchCouncil Trans. Benchmarks Stand. Eval. 1 (1) (2021) 100012.
[24] Mohamed Zairi, Paul Leonard, Origins of benchmarking and its meaning, in: Practical Benchmarking: The Complete Guide, Springer, 1996, pp. 22–27.
[25] Xingzhou Zhang, Jingjie Liu, Zhiwei Xu, Tencent and Facebook data validate Metcalfe's law, J. Comput. Sci. Tech. 30 (2) (2015) 246–251.
[26] G.M. Peter Swann, The functional form of network effects, Inf. Econ. Policy 14 (3) (2002) 417–429.
[27] Bob Briscoe, Andrew Odlyzko, Benjamin Tilly, Metcalfe's law is wrong-communications networks increase in value as they add members-but by how much? IEEE Spectr. 43 (7) (2006) 34–39.
[28] David P. Reed, That sneaky exponential—Beyond Metcalfe's law to the power of community building, Context Mag. 2 (1) (1999).

Dr. Jianfeng Zhan is a Full Professor at Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), and University of Chinese Academy of Sciences (UCAS), and director of Research Center for Advanced Computer Systems, ICT, CAS. He received his B.E. in Civil Engineering and M.Sc. in Solid Mechanics from Southwest Jiaotong University in 1996 and 1999, and his Ph.D. in Computer Science from Institute of Software, CAS, and UCAS in 2002. His research areas span from Chips, Systems to Benchmarks. A common thread is benchmarking, designing, implementing, and optimizing a diversity of systems. He has made substantial and effective efforts to transfer his academic research into advanced technology to impact general-purpose production systems. Several technical innovations and research results, including 35 patents, from his team, have been adopted in benchmarks, operating systems, and cluster and cloud system software with direct contributions to advancing the parallel and distributed systems in China or even in the world. He has supervised over ninety graduate students, post-doctors, and engineers in the past two decades. Dr. Jianfeng Zhan founds and chairs BenchCouncil and serves as the Co-EIC of TBench with Prof. Tony Hey. He has served as IEEE TPDS Associate Editor since 2018. He received the second-class Chinese National Technology Promotion Prize in 2006, the Distinguished Achievement Award of the Chinese Academy of Sciences in 2005, and the IISWC Best paper award in 2013, respectively.

Contents lists available at ScienceDirect

BenchCouncil Transactions on Benchmarks, Standards and Evaluations

journal homepage: https://www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/

Open-source computer systems initiative: The motivation, essence, challenges, and methodology

Jianfeng Zhan

Research Center for Advanced Computer Systems, Institute of Computing Technology, Chinese Academy of Sciences, China

ARTICLE INFO

Keywords:
Information technology decoupling
High-end computer systems
Opensource computer systems initiative
Challenges
Funclet
Abstraction
Architecture
Chiplet
HWlet
Envlet
Servlet

ABSTRACT

The global community faces many pressing and uncertain challenges like pandemics and global climate change. Information technology (IT) infrastructure has become the enabler to addressing those challenges. Unfortunately, IT decoupling has distracted and weakened the international community's ability to handle those challenges.

This article initiates an open-source computer system (OSCS) initiative to tackle the challenges of IT decoupling. The OSCS movement is where open-source software converges with open-source hardware. Its essential is to utilize the inherent characteristics of a class of representative workloads and propose innovative abstraction and methodology to co-explore the software and hardware design spaces of high-end computer systems, attaining peak performance, security, and other fundamental dimensions. I discuss its four challenges, including the system complexity, the tradeoff between universal and ideal systems, guaranteeing quality of computation results and performance under different conditions, e.g., best-case, worst-case, or average-case, and balancing legal, patent, and license issues.

Inspired by the philosophy of building large systems out of smaller functions, I propose the funclet abstraction and methodology to tackle the first challenge. The funclet abstraction is a well-defined, evolvable, reusable, independently deployable, and testable functionality with modest complexity. Each funclet interoperates with other funclets through standard bus interfaces or interconnections. Four funclet building blocks: chiplet, HWlet, envlet, and servlet at the chip, hardware, environment management, and service layers form the four-layer funclet architecture. The advantages of the funclet abstraction and architecture are discussed. The project's website is publicly available from https://www.opensourcecomputer.org or https://www.computercouncil.org.

1. Introduction

The complex interactions between human activities and the earth's ecosystem lead to two pressing challenges: the COVID-19 pandemic and global climate change. The study, published on 10 March in The Lancet [1], says that the actual number of lives lost to the COVID-19 pandemic by 31 December 2021 was close to 18 million. That far outstrips the 5.9 million deaths that were reported to various official sources for the same period [2]. This dire situation poses a heartbroken challenge to our seniors and children. On the other hand, due to climate change [3], more frequent and intense drought, storms, heatwaves, rising sea levels, melting glaciers, and warming oceans can directly threaten the survival of humans and wild animals.

In addition to the global society's strong support and action, the science and technology society bears the burden of tackling those challenges. Unfortunately, the growing political gaps among people with deviated viewpoints tear apart the science and technology community. Undeniably, human societies have undergone disparate political systems with varying extents of political rights and civil liberties — however, the trend converges. For example, almost every nation abolished slavery in favor of human rights; Almost every country acknowledges the rule of law, though the processes and meaning vary wildly. In the short term, there may be a spikey political gap. However, the gap has been closing for a long time. The temporally increasing political gaps do not justify the technology decoupling. Instead, technology decoupling will weaken the global community's capability to handle the pressing challenges, which adversely shakes the foundation of the worldwide community. IT is one of the enablers underpinning effective plans and actions addressing those challenges [4–7]. IT decoupling threatens the shared IT infrastructure and hence the shared future.

Technology decoupling and export control between nations will increase costs and lower productivity [8]. The dire IT decoupling distracts and weakens human beings' ability to address those pressing challenges. This calls for our wisdom and actions to unify our science and technology community. The open science initiatives [9–12,12,13] partially responded to it.

As shown in Fig. 1, this article initiates an open-source computer system movement (in short, OSCS) where open-source software converges with open-source hardware. High-end computer systems serve

E-mail address: zhanjianfeng@ict.ac.cn.
URL: http://www.benchcouncil.org/zjf.html.

https://doi.org/10.1016/j.tbench.2022.100038

Available online 25 April 2022

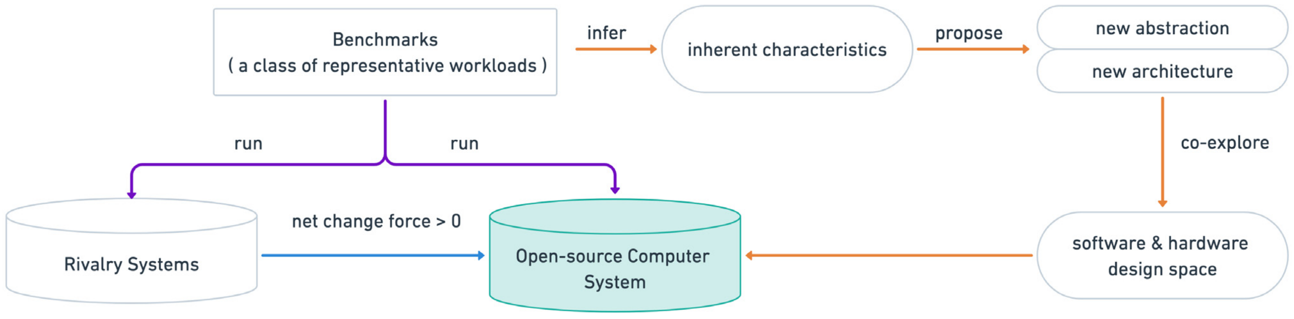


Fig. 1. The essential of the open-source computer systems (OSCS) initiative.

as the cornerstone of IT infrastructure, and their components like Chips or Operating Systems are the basis for building the IT infrastructure. The OSCS initiative chooses high-end computer systems as its target to relieve the side effect of IT decoupling. To generate a positive change force to overcome the ecosystem inertia, I use Zhan's laws of technology [8] to guide this project (The laws of technology are summarized in Section Two). On the one hand, the OSCS initiative will generate a positive change force through the open-source movement. On the other hand, it proposes an innovative methodology to improve the efficiency or other fundamental dimensions to generate the change force. The OSCS essential is to utilize the inherent characteristics of a class of representative workloads (benchmarks [14]) and propose innovative abstraction and methodology to co-explore the software and hardware design spaces of high-end computer systems, attaining peak performance, security, and other fundamental dimensions.

I discuss the four challenges of the OSCS initiative. The first challenge is the daunting system complexity witnessed by the high-end computer system and the processor ecosystem. The second challenge is how to perform the tradeoff between universal and ideal systems. For each class of representative workloads, there should be an ideal system architecture instead of a universal system where the performance, cost, or energy overhead of universality –"Turing Tax", or "Turing Tariffs" – cannot be avoided [15]. The third challenge is to propose the methodology and tools to aid the community in designing systems with guaranteed quality of computation results and performance under different conditions, e.g., best-case, worst-case, or average-case. Last but not least, it is how to balance legal, patent, and license issues of the OSCS initiative.

To tackle the first challenge, I propose the funclet abstraction and methodology. The funclet abstract represents the common proprieties of basic building blocks at different layers: each funclet is a well-defined, evolvable, reusable, independently deployable, and testable functionality with modest complexity; Each funclet interoperates with other funclets through the standard bus interfaces or interconnections. Four basic building blocks are chiplet, HWlet, envlet, and servlet at the chip, hardware, environment management, and service layers, and they form the four-layer funclet architecture. I present a three-tuple (funclet set architecture (FSA), organization, system specifics): the FSA refers to the actual programmer-visible function set [16], serving as the boundary between two adjacent layers and among different funclets in the same layer; The organization includes the high-level aspects of how funclets in the same layer and adjacent layers collaborate; The system specifics describe the design and implementation of the system built from funclets.

The structure of this article is as follows. Section Two presents the background knowledge of Zhan's three laws of technology. Section three justifies the motivation for the OSCS initiative. Section Four discusses the challenges. Section Five presents the funclet methodology. Section Six concludes.

2. Background

Zhan's three laws of technology provide a simple theoretical framework to explain and predict the rise or fall of a technology [8]. In this article, I use Zhan's three laws of technology [8] as a theoretical framework to analyze the potential and pitfall of the OSCS initiative. This section briefly introduces this framework.

The first law is on the obstacle to new technology: technology inertia. Not only end-users but also industry users stick to the existing technology, named consumer inertia and ecosystem inertia. The user size will keep constant unless a non-zero net technology change force acts on it. The second law reveals where the power of new technology comes from. The change in user size is proportional to the net technology change force. The corollary of measurement of technology change force is how to measure the net change force. By creating a brand-new technology or improving an existing technology in terms of user experience, costs, efficiency, or other fundamental dimensions by several orders of magnitude can the new technology generate a positive change force. In improving an existing technology, a new or different ecosystem will generate a negative change force $F_{Ecosystem}$, which is the side effect of ecosystem inertia. Meanwhile, different use which results in a learning cost will generate a negative change force F_{Learn}. According to the Equation in Table 1, the net change force F_t is the sum of six components: F_{Learn}, $F_{Ecosystem}$, $F_{Experience}$, F_{Cost}, $F_{Efficiency}$, F_{Other}. Table 1 summarizes the three laws and five corollaries together. Table 2 explains the symbols in the formula in Table 1.

Table 3 presents how to use Zhan's laws of technology to analyze the rise or fall of a technology, the details of which are available from [8].

3. Motivation

As shown in Fig. 2, this section explains the motivations from two perspectives: Why is IT decoupling not wise? Why launch the OSCS initiative?

3.1. Why is IT decoupling not wise?

IT infrastructure is the backbone of human society and the enabler that copes with the global pandemic and climate change challenges. For example, the scientific and engineering community heavily relies upon supercomputers to find the COVID-19 drugs and model climate change [17–19]. IT decoupling will hinder knowledge sharing and engineering collaboration, weakening our ability to handle pressing challenges. The decoupled IT communities must address severe challenges separately and amortize the unaffordable research and development costs from software and hardware.

The corollary of technology openness of Zhan's three laws of technology [8] clearly stated that contrasted with a closed ecosystem controlled by one entity, allowing the division of labor among contributors who share an open technology ecosystem improves the gross productivity and lowers the cost amortized on each contributor. Supply chain decoupling and technology export control between nations will

Table 1
The OSCS initiative uses Zhan's three laws of technology to decide the project's goal and strategy. The three laws and three corollaries used in this article are summarized while omitting the other two corollaries. The full details are shown in [8].

Law or Corollary name	Formula
Law of technology inertia	$\Delta U_t = U_{(t+\Delta t)} - U_t$ $\Delta U_t = 0$, U *is a natural number*
Law of technology change force	$\Delta U_t \propto F_t$, U *is a natural number*
Law of technology change action and reaction	$F_{Emerging} = -F_{Existing}$
Corollary of measurement of technology change force	$F_t = F_{Create} + F_{Learn} + F_{Ecosystem}$ or $F_t = F_{Experience} + F_{Cost} + F_{Efficiency} + F_{Other} + F_{Learn} + F_{Ecosystem}$
Corollary of technology breakthrough	$B = F/U$, U *is a natural number*
Corollary of technology openness	$P = p * U_i,\ c = C/U_i$ $P = p * U_i/M,\ c = C * M/U_i$ $P = p * U_i/N,\ c = C * N/U_i$

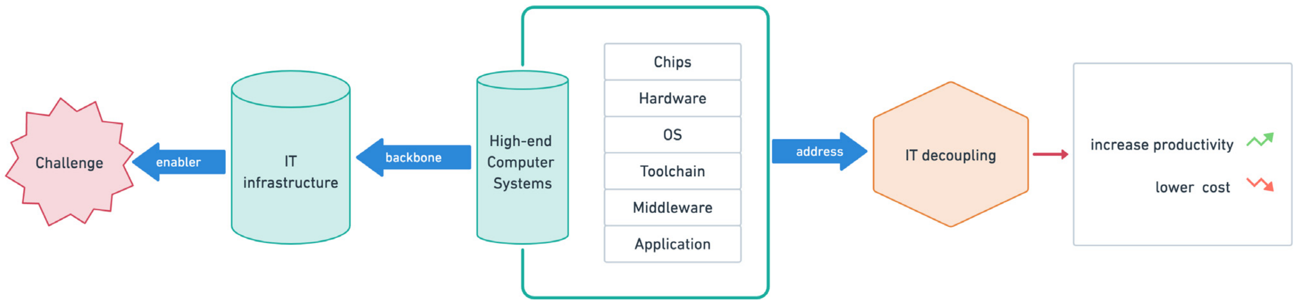


Fig. 2. The motivation for launching the open-source computer systems (OSCS) initiative.

Table 2
The explanations of symbols in Table 1 [8].

Symbol	Explanation
Δ	Difference operator
$\propto$	Proportional operator
Σ	Summation operator
U_t	User size varying with time
U_i	Size of industry users
F_t	Net technology change force varying with time
$F_{Emerging}$	Change force acting on emerging technology
$F_{Existing}$	Change force acting on existing technology
F_{Create}	Change force resulted from creating a brand-new technology
F_{Learn}	Change force resulted from learning cost
$F_{Ecosystem}$	Change force resulted from ecosystem deviation
$F_{Experience}$	Technology change force resulted from user experience
F_{Cost}	Change force resulted from cost
$F_{Efficiency}$	Change force resulted from efficiency
F_{Other}	Change force resulted from other fundamental dimensions
B	Technology breakthrough
P	Gross productivity
C	Total cost
M	Number of decoupled supply chains
N	Number of nations
c	Cost
p	Productivity of each contributor (industry user)

increase costs and lower productivity. The dire IT decoupling will distract and weaken human beings' ability to handle those pressing challenges. This calls for our wisdom and actions to unify our science and technology community. The open science initiatives [9–12,12,13] partially responded to it.

3.2. Why launch the OSCS initiative?

The open-source software movement has become mainstream, like closed-source ones. The open-source software outspring includes Linux, Android, and many other software stacks. Opensource hardware is sporadic with a handy of hardware components that attain the performance and reliability that amount to the commodity components. However, it is far from ready to handle IT decoupling challenges.

First, as demonstrated in Table 4, IT infrastructure like high-end computer systems kept closed even open-source movement makes excellent progress. As shown in Fig. 2, high-end computer systems not only serve as the cornerstone of the IT infrastructure, but its components, like chips, hardware, OS, toolchain, and middleware, are also the basis for building IT infrastructure. So high-end computer systems are vital to addressing the challenges of IT decoupling.

The Open Compute Project Foundation (OCP) [30] was initiated in 2011 with a mission to open-source datacenter hardware (warehouse-scale computing) in mind; it still makes little progress in essential components. RISC-V [31] is an open standard instruction set architecture (ISA) following reduced instruction set computer (RISC) principles. The open-source chip project like RISC-V is promising as it is provided under open source licenses that do not require fees to use, unlike most other ISA designs [31]. For example, Institute of Computing Technology at Chinese Academy of Sciences has showcased progress on a fully open-source RISC-V processor, XiangShan, or "Fragrant Hills" [32], which is promising in competing Arm counterparts.

Second, fundamental changes in technology favor domain-specific hardware and software co-design, e.g. end of Dennard scaling, ending of Moore's Law, Amdahl's Law and its implications for ending 'easy' multicore era [33,34]. In the new golden age of computer architecture, it is essential to consider software and hardware together. So it is time for opensource software movements to converge with opensource hardware movements.

Third, merely repeating the methodology and process and replicating the closed-source counterpart will not succeed directly. According to the law of technology change force [8], the open-source initiative will make $F_{Cost} > 0$. Improving the closed-source counterpart in terms of user experience, costs, efficiency, or other fundamental dimensions, the open-source one can generate other positive components of the change force, e.g., $F_{Experience}$ – technology change force resulted from user experience, $F_{Efficiency}$ – change force resulted from efficiency, F_{Other} – change force resulted from other fundamental dimensions.

Table 3
The analysis of thirteen successful IT using Zhan's three laws of technology [8]. The essential of using those laws is to measure the components of change force. The net change forces decide each technology's rise or fall. Contrasted with a closed ecosystem controlled by one entity, allowing the division of labor among contributors sharing an open technology ecosystem improves the gross productivity and lowers the costs amortized on each contributor [8]. Decoupling the supply chain will increase costs and lower productivity [8].

Technology	Rivals	F_{Learn}	$F_{Ecosystem}$	F_{Create}	F_{Cost}	$F_{Efficiency}$	$F_{Experience}$	F_{Other}	Closed/Open	Supply Chain
Deep learning	Shallow neutral networks	0	<0	/	<0	/	/	≫0 (accuracy)	Open	Coupling
WWW	No	<0	<0	≫0	/	/	/	/	Open	Coupling
Google	No	<0	<0	≫0	/	/	/	/	Closed	Decoupling
Facebook	No	<0	<0	≫0	/	/	/	/	Closed	Decoupling
Internet	No	<0	<0	≫0	/	/	/	/	Open	Coupling
RAID	Single large expensive disk	0	0	/	>0	>0	/	/	Open	Coupling
Android	Windows Mobile, Symbian, iOS, Linux	<0	0	/	>0	/	>0	>0 (Google ecosystem)	Open	Coupling
iOS	Windows Mobile, Symbian	<0	<0	/	<0	/	≫0	>0 (AppStore)	Closed	Coupling
Windows	DOS	<0	0	/	/	/	>0	/	Closed	Coupling
Linux	UNIX	0	0	/	>0	/	/	/	Open	Coupling
UNIX	Multics	<0	<0	/	/	>0	/	>0 (standard)	Closed	Coupling
ARM	X86, RISC	0	0	/	/	>0	/	>0 (energy efficiency)	Closed	Decoupling
RISC	CISC	0	0	/	/	>0	/	/	Closed	Decoupling

Table 4
Eight categories of high-end computer systems.

Domain	Benchmark [14]	Metrics	Status	OSCS target
Planet-scale computers (Distributed IoTs, Edges, and datacenter systems) [20]	ScenarioBench [21]	Undefined	Not yet mature	Yes
AI for science	SAIBench [22]	Undefined	Not yet mature	Yes
Deep learning	AIBench [23] or MLPerf [24,25]	State-of-the-quality	Mature	No
Metaverse	MetaverseBench [26]	N/A	Not yet mature	Yes
High performance computing	HPCC [27]	FLOPS	Mature	No
Warehouse-scale computing	N/A	Throughput, Tail latency	Mature	No
Big Data	BigDataBench [28] or BigBench [29],	Throughput, Quality of services, turnaround	Mature	No
Cloud computing	N/A	System utilization, Quality of services	Mature	NO

It is necessary to take other actions to generate other positive components of the change force. On the one hand, it is essential to leverage the inherent characteristics of a class of representative workloads (The second category of benchmarks in [14]) to co-explore the software and hardware architecture space [33,34]. On the other hand, it is necesary to develop new abstraction, methodology, and architecture in the OSCS initiative.

Fourth, being compatible with the ecosystem and users' learning habits is essential. Not only end-users but also industry users stick to the existing technology, which is consumer inertia and ecosystem inertia [8]. According to the corollary of measurement of technology change Force of Zhan's laws of technology, A new ecosystem or the deviation from existing technology ecosystems will generate a negative change force $F_{Ecosystem}$; Different use, which results in an end-user learning cost, will generate a negative change force F_{Learn}.

I conclude the essence of the OSCS initiative. The OSCS initiative has four implications. (1) It is an open-source movement where software converges with hardware. (2) It is to utilize the inherent characteristics of a class of representative workloads (The second category of benchmarks in [14]). (3) Instead of re-inventing the wheel, it is to propose innovative abstraction and methodology to co-explore the software and hardware design space to attain peak performance, security, and other fundamental dimensions. (4) it emphasizes compatibility with the ecosystem and users' learning habits. Fig. 1 reveals the essence of the OSCS initiative visually.

According to the Corollary of measurement of technology change force, the goal of the strategy is to maximize the positive value of the net change force. High-performance computing, cloud computing, and warehouse-scale computing are mature. Hence, generating a net change force that breaks through the technology inertia is much more challenging, i.e., improving an existing technology in terms of user experience, costs, efficiency, or other fundamental dimensions by several orders of magnitude. So I choose three emerging areas: planet-scale computers, which redesign the IoTs, edges, data centers and networks as a computer [20], AI for sciences, and Metaverse as the initial three targets of the OSCS initiative. I enforce the following strategies for each class of systems to maximize the net change force. (1) open-source initiative, which makes $F_{Cost} > 0$; (2) sharpen the edges like efficiency ($F_{Efficiency} > 0$, and the other fundamental dimension $F_{Other} > 0$; (3) provide the compatible ecosystem and lower the learning cost. ($F_{Learn} = 0$, $F_{Ecosystem} = 0$).

4. The challenges

This subsection will present the high-level challenges of the OSCS initiative. Low-level challenges are thoroughly discussed in [20,22,26].

(1) The challenge of system complexity.

I demonstrate the system complexity from two dimensions: the high-end computer systems and the processor ecosystem.

As shown in Fig. 2, high-end computer systems are the cornerstone of IT infrastructure with daunting complexity. As a case study, I review the state-of-the-art supercomputers on the Top 500 list [35]. Fugaku held the No. 1 position that it first earned in June 2020. Fugaku is based on Fujitsu's custom ARM A64FX processor, each with four NUMA nodes. With each NUMA node having 12 compute cores, each processor has 48 cores. Fugaku has 7,630,848 cores – using Fujitsu's Tofu-D interconnect to transfer data between nodes– achieving an HPL benchmark score of 442 Pflop/s [35]. High-end computer systems are a vivid demonstration of system engineering and art.

As I earlier analyzed in [8], an entire ecosystem of X86, ARM, or RISC-V processors consists of SoC (a system on a chip), ISA (Instruction Set Architecture), OS (operating system), toolchain, middleware, and applications. It is far beyond the reach of state-of-the-art and state-of-the-practice open-source projects. Even considering only the components, modern systems like Systems on Chips (SoCs) or Operating Systems have growing complexity that leads to a design productivity crisis [36]. For the SoC, the chipmaker shrinks different functions at each node and packs them onto a monolithic die, which becomes more complex and expensive at each node [37].

The high-end computer systems in Table 4 require aggressive, tactful, and coordinated plans for open-source computer systems.

(2) The tradeoff challenges between universal and ideal systems.

Turing [38] proposed the idea of a "universal" computing device – a single device that can implement any computable function. However, it will introduce "Turing Tax", or "Turing Tariffs" [15]: the performance, cost, or energy overhead of universality — the difference between a special-purpose device and a general-purpose one. Because industrial users adhere to existing products, tools, platforms, and services for investment protection — so-called ecosystem inertia [8], the user tends to opt for the universal systems or much narrow-scope general-purpose systems like GPGPUs. Modern systems, e.g., Systems on chip (SoCs) or Operating Systems, have growing complexity that leads to a design productivity crisis [36], and the communities cannot afford the cost of building new systems. That is another reason users prefer the universal or much narrower-scope general-purpose systems.

However, the new technology trends have gained momentum. The agile software and hardware process and methodology [36,39] enables small-team to build competitive high-performance microprocessors and software systems quickly. For each class of representative workloads, there should be an ideal system architecture instead of a universal system. Single-purpose system is not preferred; however, it is reasonable to tradeoff between the universal and ideal systems. Specifically, it motivates each class of workloads to explore the ideal architecture space.

(3) The challenges of guaranteeing the quality of computation results and performance under best-case, worst-case, or average-case.

Modern systems care about performance and value the quality of their computation results. For example, the deep learning community has come to terms with accepting the time of training an AI model to achieve a state-of-the-art quality as the primary metrics [23,25,40] in evaluating the system, which values both systems and algorithms.

I think the community should consider the quality of computation results and performance under different cases like best-case, worst-case, or average-case as first-class constraints in system design, implementation, verification,[1] and validation.[2] For example, how does a system achieve state-of-the-art quality with the 99th tail latency of fewer than 100 milliseconds? How to estimate or guarantee the performance and quality that satisfy a threshold in the best case (1 out of 1000) – consider the case of searching for alien civilizations? How about the average case? The design, implementation, verification, and validation costs vary wildly in different cases. The designers need to propose a new methodology to design, implement, verify and validate the systems with guaranteed confidence in quality and performance. Proposing the appropriate metrics is the first step in any-case system challenges. It is pressing to propose a new design, implementation, verification, and validation methodology and tool to address any-case system challenges.

I do not mean that these issues were totally overlooked in the past. HPC communities have much expertise in the tradeoff between performance and accuracy. Langou et al. [41] exploit single-precision operations whenever possible and resort to double-precision at critical stages while attempting to provide the full double precision results. The real-time system community [42] considered the hard and soft deadlines as first-class design constraints. In warehouse-scale computing [43], the tail latency of large-scale internet service – worst-case latency expressed in a percentile term, e.g., 98th, 99th – becomes the primary metric that outweighs the average latency. Lu et al. [44] claimed the modern data center operating system should gracefully achieve disparate performance goals in terms of both average and worst-case performance.

(4) The challenges of balancing legal, patent, and license issues.

In the past several decades, many different software models have become mainstream, such as the proprietary software product vendor model, the custom software developer model, advertising-supported software, and web-delivered software as a service [45]. How the OSCS initiative interacts and impacts those models? The license and patent policy may significantly impact that process. The situation becomes much more complex, especially considering the potential issues that may come to courts with the commercial project that hybridizes proprietary and open-source mechanisms [45]. Legal issues like long-arm jurisdiction in export control further complicate these situations.

The OSCS initiative is not a perfect plan. Also, it has several side effects. The most prominent one is how to prevent extremists or terrorism from leveraging open-source high-end computer systems.

5. The funclet methodology

This section presents the methodology for tackling the first challenge. I left the other challenges in the future work.

When the IT pioneer Gordon Moore [46] envisioned the future, he concluded "it might prove to be more economical to build large systems out of smaller functions, which are separately packaged and interconnected". Inspired by this philosophy of building large systems out of smaller functions, I propose the funclet methodology. First, I present the funclet abstraction, then the funclet architecture.

The funclet abstract represents the common proprieties of basic building blocks at different layers. Each funclet has the following characteristics. (1) each contains a well-defined and evolvable functionality with modest complexity (2) each can be reusable in different contexts. (3) each can be independently tested and verified before integrating. (4) each can be independently deployable. (5) each can interoperate with other funclets through a well-defined bus interface or interconnection.

As shown in Fig. 3, I propose a four-layer funclet architecture. As a start, I reuse two emerging concepts to describe funclet at the first and fourth layers, and then I elaborate on the other layers. The first-layer funclet is a chiplet, which is an integrated circuit (IC) with modest complexity, providing well-defined functionality [37,47]; it is designed to be susceptible to integration with other chiplets, connected with a die-to-die interconnect scheme [37,47]. A chiplet differs from the traditional, monolithic system on chip (SoC) in the following way [37,47]: a chipmaker can mix and match chiplets to reduce product development times and costs by integrating pre-developed die in an IC package [37, 47].

The fourth-layer funclet is a servlet, an independently deployable and evolvable component that severs users with a well-defined and modest-complexity functionality. A servlet supports interoperability through standardized software bus [48]. Microservice [48] is a form of a servlet. Another related concept is cloud functions [49]. Cloud functions package as Function as a Service (FaaS) offerings [49], which represents the core of serverless computing. Cloud functions are the general-purpose elements in serverless computing, leading to a simplified and general-purpose cloud programming model [49].

The second-layer funclet is an HWlet, an independently deployable, replaceable, and accessible hardware component, e.g., CPU, memory, storage. An HWlet could be aggregated into a resource pool with low latency and high bandwidth interconnection. I explain the HWlet concept in the context of an aggregated architecture [50–52]. For example, an HWlet could be a commodity memory module in a disaggregated memory design. Multiple compute blades can access an array of commodity memory modules that are encapsulated in a separate shared memory blade via a shared blade interconnect [52].

The third-layer funclet is an envlet, which is an independently deployable and evolvable environment component with well-defined functionality that supports the management of servlets. Envlets can interoperate through interconnections to form a management infrastructure. In the context of serverless computing, BaaS (Backend as a Service) [49] is a form of envlet. BasS provides the management services for cloud functions, responsible for the latter's automatic scaling with no need for explicit provisioning and usage-based billing [49].

[1] Verification determines whether each component and the assembly of components correctly meets its specification through testing("Are we building the thing right?") [36].

[2] Validation ensures that the product serves its intended purposes ("Are we building the right thing?") [36].

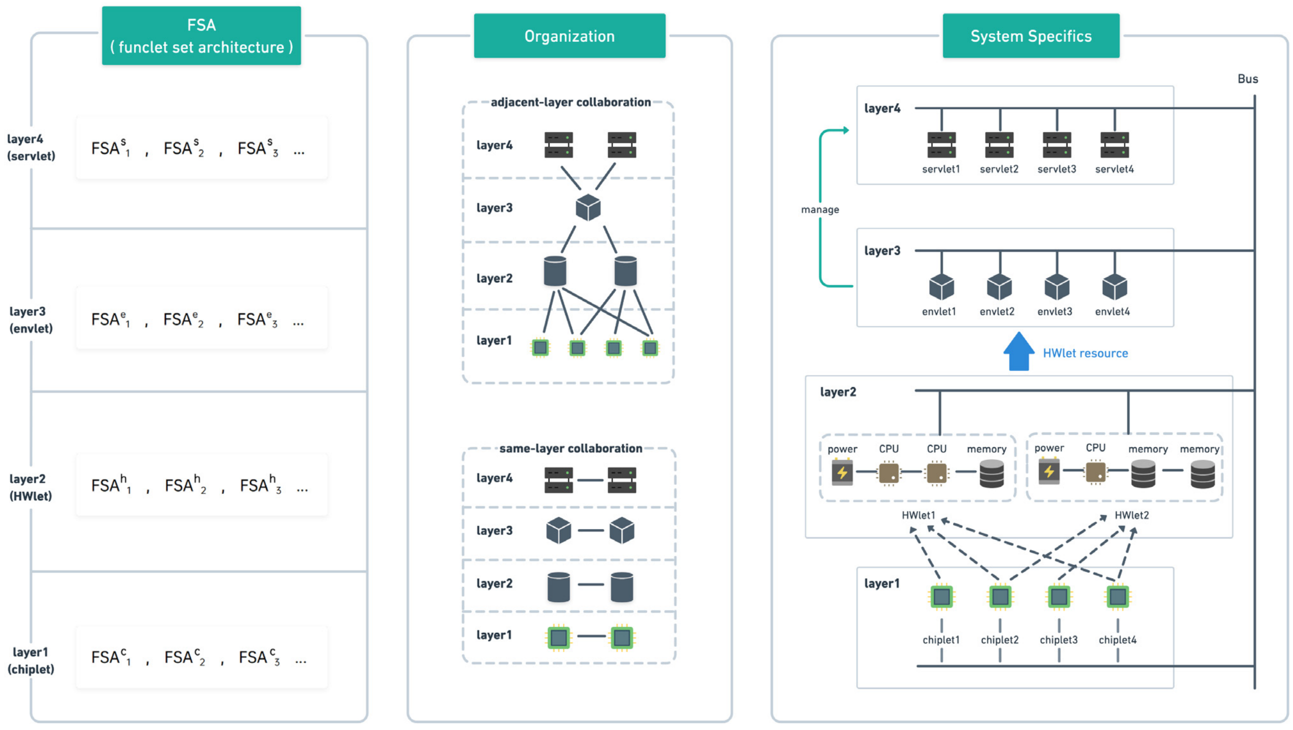


Fig. 3. An instance of the four-layer funclet architecture.

I use a methodology inspired by the computer architecture community to explicitly specify how funclets work together. I first present the methodology from the computer architecture community [16]. Then I elaborate on my proposed methodology.

In computer architecture, the terms instruction set architecture (ISA), organization or microarchitecture, and hardware are used to describe the architecture — the computer's design and implementation [16]. The ISA refers to the actual programmer-visible instruction set, serving as the boundary between the software and hardware [16]. The organization includes the high-level aspects of a computer's design, such as the memory system, the memory interconnects, and the design of the internal processor or CPU [16]. The hardware refers to the specifics of a computer, including the detailed logic design and the packaging technology of the computer [16].

I use a three-tuple {funclet set architecture (FSA), organization, system specifics} methodology to describe the funclet architecture, as shown in Fig. 3. The FSA refers to the actual programmer-visible function set [16], serving as the boundary between two adjacent layers and among different funclets in the same layer. The organization includes the high-level aspects of how funclets in the same layer and adjacent layers collaborate. The system specifics describe the design and implementation of a system built from funclets.

The funclet methodology specifies the architecture space in terms of (FSA, organization, system specifics). There is an explosive architecture space when searching for the optimal design. There is a pressing need for tools aiding the exploration of the funclet architecture. As each chiplet, HWlet, envlet, and servlet provide narrow-scoped functionality, it is essential to align the functions of the funclets that have similar resource-consumption characteristics for whole-stack optimization and resource management. In this context, the implication of the open-source computer systems initiative is to utilize the inherent characteristics of a class of representative workloads to co-explore the hardware and software design space in terms of (FSA, organization, system specifics) to attain peak performance and security.

I call the traditional system architecture, which consists of a monolithic chip, hardware, management environment, and application or services — the monolithic architecture. Meanwhile, some architectures have partially used the chiplet, microservice, or cloud functions, which I call hybrid architecture.

The advantage of the funclet architecture is four-fold. (1) it increases technology openness, improves productivity, and lowers cost. The funclet philosophy and methodology facilitate the contributors to focus on each funclet, allowing the efficient division of labor among contributors sharing an open technology ecosystem [8]. It finally improves the gross productivity and lowers the cost amortized on each contributor [8]. (2) it can help tackle the challenge of the complexity of computer systems from the horizontal and vertical dimensions. The system is divided among chiplet, HWlet, Envlet, and servlet from the vertical dimension. The system is described from FSA, organization, and system specifics from the horizontal and vertical dimensions. (3) it can help optimize the whole-stack system. With the close alignment of the functions of a servlet, Envlet, HWlet, and chiplet, the system can be optimized across the whole stack for narrow-scoped workloads with similar characteristics, e.g., resource requirements. (4) it can improve the reusability. Each funclet can be independently designed, implemented, and tested. Finally, the users can assemble the funclets into a complex system. (5) it can improve reliability. Before assembling, each funclet can be independently deployed and tested, enhancing the whole system's reliability.

Here, I emphasize the goal of the funclet architecture is not to replace all the monolithic or hybrid architectures. Instead, the former complements the latter. Similarly, Zhan's three laws of technology [8] will govern how they compete in the future.

6. Conclusion

This article initiates an open-source computer system (OSCS) movement. The OSCS initiative is an open-source movement where software converges with hardware. It is to utilize the inherent characteristics of a class of representative workloads and propose innovative abstraction and methodology to co-explore the software and hardware design space to attain peak performance, security, and other fundamental dimensions. I discuss four OSCS challenges: the daunting system complexity, the tradeoff between universal and ideal systems, guaranteeing best-case, worst-case, or average-case quality and performance, and balancing legal, patent, and license issues.

I propose the funclet abstraction and architecture to tackle the system complexity challenges. The funclet abstraction is a well-defined, evolvable, reusable, independently deployable, and testable functionality with modest complexity. Each funclet interoperates with other funclets through standard bus interfaces or interconnections. Four funclet building blocks: chiplet, HWlet, envlet, and servlet at the chip, hardware, environment management, and service layers form the four-layer funclet architecture.

Acknowledgments

I am very grateful to Mr. Shaopeng Dai and Mr. Qian He for drawing Figs. 1, 2, and 3, Dr. Lei Wang for contributing to Fig. 3 and proofreading the article, Dr. Wanling Gao for proofreading this article.

References

[1] C.-.E.M. Collaborators, Estimating excess mortality due to the COVID-19 pandemic: a systematic analysis of COVID-19-related mortality, 2020–21, Lancet 21 (6) (2022) 691–708.
[2] D. Adam, Covid's true death toll: much higher than official records, 2022, https://www.nature.com/articles/d41586-022-00708-0.
[3] WWF, IMPACTS of Global Climate Change.
[4] Engineering, and Medicine and others National Academies of Sciences, Information Technology Innovation: Resurgence, Confluence, and Continuing Impact, National Academies Press, 2020.
[5] National Research Council and others, Continuing Innovation in Information Technology, The National Academies Press, 2012.
[6] National Research Council, Innovation in Information Technology, The National Academies Press, 2003.
[7] National Research Council and others, Funding a Revolution: Government Support for Computing Research, National Academies Press, 1999.
[8] J. Zhan, Three laws of technology rise or fall, BenchCouncil Trans. Benchmarks Stand. Eval. (2022) 100034.
[9] S. Friesike, T. Schildhauer, Open science: many good resolutions, very few incentives, yet, in: Incentives and Performance, Springer, 2015, pp. 277–289.
[10] A.B. Powell, Open culture and innovation: integrating knowledge across boundaries, Media, Culture Soc. 37 (3) (2015) 376–393.
[11] J.M. Pearce, Open-Source Lab: How To Build Your Own Hardware and Reduce Research Costs, Newnes, 2013.
[12] A. Katz, Towards a functional license for open hardware, IFOSS L. Rev. 4 (2012) 41.
[13] P.A. David, Towards a cyberinfrastructure for enhanced scientific collaboration: providing its' soft'foundations may be the hardest part, 2004.
[14] J. Zhan, Call for establishing benchmark science and engineering, BenchCouncil Trans. Benchmarks Stand. Eval. 1 (1) (2021) 100012.
[15] P.H.J. Kelly, "Turing tariff" reduction: architectures, compilers and languages to break the universality barrier, 2020, https://www.doc.ic.ac.uk/~phjk/Presentations/2020-06-24-DoCLunch-PaulKelly-TuringTaxV04.pdf.
[16] J.L. Hennessy, D.A. Patterson, Computer Architecture: A Quantitative Approach, Elsevier, 2019.
[17] Z. Shervani, I. Khan, T. Khan, U.Y. Qazi, et al., World's fastest supercomputer picks COVID-19 drug, Adv. Infect. Dis. 10 (03) (2020) 211.
[18] D. Adam, Simulating the pandemic: What COVID forecasters can learn from climate models, Nature 587 (7835) (2020) 533–535.
[19] H. Fu, J. Liao, J. Yang, L. Wang, Z. Song, X. Huang, C. Yang, W. Xue, F. Liu, F. Qiao, et al., The sunway TaihuLight supercomputer: system and applications, Sci. China Inf. Sci. 59 (7) (2016) 1–16.
[20] ComputerCouncil, The IoTs, edges, datacenter and networks as a computer: Building open-source planet-scale computers (PSC) for emerging and future computing, 2022, https://www.computercouncil.org/PSC.
[21] W. Gao, F. Tang, J. Zhan, X. Wen, L. Wang, Z. Cao, C. Lan, C. Luo, X. Liu, Z. Jiang, Aibench scenario: Scenario-distilling ai benchmarking, in: 2021 30th International Conference on Parallel Architectures and Compilation Techniques (PACT), IEEE, 2021, pp. 142–158.
[22] Y. Li, J. Zhan, SAIBench: Benchmarking AI for science, BenchCouncil Trans. Benchmarks Stand. Eval. (2022) 100034.
[23] F. Tang, W. Gao, J. Zhan, C. Lan, X. Wen, L. Wang, C. Luo, Z. Cao, X. Xiong, Z. Jiang, et al., Aibench training: balanced industry-standard ai training benchmarking, in: 2021 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), IEEE, 2021, pp. 24–35.
[24] P. Mattson, C. Cheng, G. Diamos, C. Coleman, P. Micikevicius, D. Patterson, H. Tang, G.-Y. Wei, P. Bailis, V. Bittorf, et al., Mlperf training benchmark, Proc. Mach. Learn. Syst. 2 (2020) 336–349.
[25] Y.-H. Chang, J. Pu, W.-m. Hwu, J. Xiong, Mlharness: A scalable benchmarking system for mlcommons, BenchCouncil Trans. Benchmarks, Stand. Eval. 1 (1) (2021) 100002.
[26] ComputerCouncil, Metaversebench: Instantiating and quantifying metaverse problems, benchmarks, and challenges, 2022, https://www.computercouncil.org/MetaverseBench.
[27] P.R. Luszczek, D.H. Bailey, J.J. Dongarra, J. Kepner, R.F. Lucas, R. Rabenseifner, D. Takahashi, The HPC Challenge (HPCC) benchmark suite, in: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing, Vol. 213, 10.1145, pp. 1188455–1188677.
[28] L. Wang, J. Zhan, C. Luo, Y. Zhu, Q. Yang, Y. He, W. Gao, Z. Jia, Y. Shi, S. Zhang, et al., Bigdatabench: A big data benchmark suite from internet services, in: 2014 IEEE 20th International Symposium on High Performance Computer Architecture (HPCA), IEEE, 2014, pp. 488–499.
[29] A. Ghazal, T. Rabl, M. Hu, F. Raab, M. Poess, A. Crolotte, H.-A. Jacobsen, Bigbench: Towards an industry standard benchmark for big data analytics, in: Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data, 2013, pp. 1197–1208.
[30] The Open Compute Project Foundation, The open compute project, 2022, https://www.opencompute.org/.
[31] A. Waterman, Y. Lee, D.A. Patterson, K. Asanovi, The risc-v instruction set manual. volume 1: User-level isa, version 2.0, Technical Report, California Univ Berkeley Dept of Electrical Engineering and Computer Sciences, 2014.
[32] Gareth Halfacree, Chinese chip designers hope to topple arm's cortex-a76 with XiangShan RISC-v design, 2021, https://www.theregister.com/2021/07/06/xiangshan_risc_v/.
[33] J.L. Hennessy, D.A. Patterson, A new golden age for computer architecture, Commun. ACM 62 (2) (2019) 48–60.
[34] W. Gao, J. Zhan, L. Wang, C. Luo, D. Zheng, F. Tang, B. Xie, C. Zheng, X. Wen, X. He, et al., Data motifs: A lens towards fully understanding big data and ai workloads, in: Proceedings of the 27th International Conference on Parallel Architectures and Compilation Techniques, 2018, pp. 1–14.
[35] P.H.J. Kelly, Still waiting for exascale: Japan's fugaku outperforms all competition once again, 2021, https://www.top500.org/news/still-waiting-exascale-japans-fugaku-outperforms-all-competition-once-again/.
[36] Y. Lee, A. Waterman, H. Cook, B. Zimmer, B. Keller, A. Puggelli, J. Kwak, R. Jevtic, S. Bailey, M. Blagojevic, et al., An agile approach to building RISC-V microprocessors, Ieee Micro 36 (2) (2016) 8–20.
[37] Mark Lapedus, The good and bad of chiplets, 2020, https://semiengineering.com/the-good-and-bad-of-chiplets/.
[38] R. Herken, The Universal Turing Machine a Half-Century Survey, Springer-Verlag, 1995.
[39] M. Fowler, J. Highsmith, et al., The agile manifesto, Softw. Develop. 9 (8) (2001) 28–35.
[40] C. Coleman, D. Narayanan, D. Kang, T. Zhao, J. Zhang, L. Nardi, P. Bailis, K. Olukotun, C. Ré, M. Zaharia, Dawnbench: An end-to-end deep learning benchmark and competition, Training 100 (101) (2017) 102.
[41] J. Langou, J. Langou, P. Luszczek, J. Kurzak, A. Buttari, J. Dongarra, Exploiting the performance of 32 bit floating point arithmetic in obtaining 64 bit accuracy (revisiting iterative refinement for linear systems), in: SC'06: Proceedings of the 2006 ACM/IEEE Conference on Supercomputing, IEEE, 2006, p. 50.
[42] P.A. Laplante, et al., Real-Time Systems Design and Analysis, Wiley New York, 2004.
[43] L.A. Barroso, U. Hölzle, The datacenter as a computer: An introduction to the design of warehouse-scale machines, Synthesis Lect. Comput. Arch. 4 (1) (2009) 1–108.
[44] G. Lu, J. Zhan, C. Tan, X. Lin, D. Kong, C. Zheng, F. Tang, C. Huang, L. Wang, T. Hao, Isolate first, then share: a new os architecture for datacenter computing, 2016, arXiv preprint arXiv:1604.01378.
[45] G.R. Vetter, Commercial free and open source software: knowledge production, hybrid appropriability, and patents, Fordham L. Rev. 77 (2008) 2087.
[46] G.E. Moore, et al., Cramming more components onto integrated circuits, 1965.
[47] T. Li, J. Hou, J. Yan, R. Liu, H. Yang, Z. Sun, Chiplet heterogeneous integration technology—Status and challenges, Electronics 9 (4) (2020) 670.
[48] I. Nadareishvili, R. Mitra, M. McLarty, M. Amundsen, Microservice architecture: aligning principles, practices, and culture, " O'Reilly Media, Inc.", 2016.
[49] E. Jonas, J. Schleier-Smith, V. Sreekanti, C.-C. Tsai, A. Khandelwal, Q. Pu, V. Shankar, J. Carreira, K. Krauth, N. Yadwadkar, et al., Cloud programming simplified: A berkeley view on serverless computing, 2019, arXiv preprint arXiv:1902.03383.
[50] J. Fan, M. Chen, Dynamic self-organized computer architecture based on grid-components(DSAG), J. Comput. Res. Develop. 40 (12) (2003) 1737–1742.
[51] K. Lim, J. Chang, T. Mudge, P. Ranganathan, S.K. Reinhardt, T.F. Wenisch, Disaggregated memory for expansion and sharing in blade servers, ACM SIGARCH Comput. Archit. News 37 (3) (2009) 267–278.
[52] K. Lim, Y. Turner, J.R. Santos, A. AuYoung, J. Chang, P. Ranganathan, T.F. Wenisch, System-level implications of disaggregated memory, in: IEEE International Symposium on High-Performance Comp Architecture, IEEE, 2012, pp. 1–12.

Dr. **Jianfeng Zhan** is a Full Professor at Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), and University of Chinese Academy of Sciences (UCAS), the director of Research Center for Advanced Computer Systems, ICT, CAS. He received his B.E. in Civil Engineering and MSc in Solid Mechanics from Southwest Jiaotong University in 1996 and 1999, and his Ph.D. in Computer Science from Institute of Software, CAS, and UCAS in 2002. His research areas span from Chips, Systems to Benchmarks. A common thread is benchmarking, designing, implementing, and optimizing a diversity of systems. He has made substantial and effective efforts to transfer his academic research into advanced technology to impact general-purpose production systems. Several technical innovations and research results, including 35 patents, from his team, have been adopted in benchmarks, operating systems, and cluster and cloud system software with direct contributions to advancing the parallel and distributed systems in China or even in the world. He has supervised over ninety graduate students, post-doctors, and engineers in the past two decades. Dr. Jianfeng Zhan founds and chairs BenchCouncil and serves as the Co-EIC of TBench with Prof. Tony Hey. He has served as IEEE TPDS Associate Editor since 2018. He received the second-class Chinese National Technology Promotion Prize in 2006, the Distinguished Achievement Award of the Chinese Academy of Sciences in 2005, and the IISWC Best paper award in 2013, respectively.

Contents lists available at ScienceDirect

BenchCouncil Transactions on Benchmarks, Standards and Evaluations

journal homepage: https://www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/

Are current benchmarks adequate to evaluate distributed transactional databases?

Luyi Qu [a], Qingshuai Wang [a], Ting Chen [a], Keqiang Li [a], Rong Zhang [a,*], Xuan Zhou [a], Quanqing Xu [b], Zhifeng Yang [b], Chuanhui Yang [b], Weining Qian [a], Aoying Zhou [a]

[a] East China Normal University, China
[b] OceanBase, China

ARTICLE INFO

Keywords:
Distributed transactional databases
Benchmarks
Workload

ABSTRACT

With the rapid development of distributed transactional databases in recent years, there is an urgent need for fair performance evaluation and comparison. Though there are various open-source benchmarks built for databases, it is lack of a comprehensive study about the applicability for distributed transactional databases. This paper presents a review of the state-of-art benchmarks with respect to distributed transactional databases. We first summarize the representative architectures of distributed transactional databases and then provide an overview about the chock points in distributed transactional databases. Then, we classify the classic transactional benchmarks based on their characteristics and design purposes. Finally, we review these benchmarks from schema and data definition, workload generation, and evaluation and metrics to check whether they are still applicable to distributed transactional databases with respect to the chock points. This paper exposes a potential research direction to motivate future benchmark designs in the area of distributed transactional databases.

Contents

* Corresponding author.
E-mail addresses: luyiqu@stu.ecnu.edu.cn (L. Qu), qswang@stu.ecnu.edu.cn (Q. Wang), tingc@stu.ecnu.edu.cn (T. Chen), kqli@stu.ecnu.edu.cn (K. Li), rzhang@dase.ecnu.edu.cn (R. Zhang), xzhou@dase.ecnu.edu.cn (X. Zhou), xuquanqing.xqq@oceanbase.com (Q. Xu), zhuweng.yzf@oceanbase.com (Z. Yang), rizhao.ych@oceanbase.com (C. Yang), wnqian@dase.ecnu.edu.cn (W. Qian), ayzhou@dase.ecnu.edu.cn (A. Zhou).

https://doi.org/10.1016/j.tbench.2022.100031
Received 30 September 2021; Received in revised form 15 January 2022; Accepted 15 January 2022
Available online 10 February 2022

1. Introduction

Though the traditional stand-alone relational database management system (*RDBMS*) has attracted great attention, it is still limited in the scalability of storage and computing for large application scenarios, e.g., *Securities Exchange*. Business expansion or new application emerging with a characteristic of high throughput or large storage promotes the designing and developing of distributed databases, which have become a hot topic in both academia and industry [1–13]. During the long exploration process from the stand-alone database to the distributed one, various solutions are coming up with respect to different application requirements. For example, in E-commerce [14], it may separate $Read$ from $Write$ to improve throughputs ; when meeting hotspots, it may split data partition or move a part to a free node. All the effort is to realize database scalability on different implementation modules. The representative distributed databases are arranged along the timeline by either the paper published time or the project opensourced time in github (tagged by g) in Fig. 1. Notice that, the earliest opensourced version of *OceanBase* is published in 2014, but its architecture is quite different from the current one. Therefore, we put the latest one here.

NoSQL databases [15–17] firstly design many ways to leverage distributed storages and computational power from multiple machines, which are widely used for web applications. These businesses have critical requirements for large storage and concurrent processing. However, due to the lack of *SQL* compatibility and *ACID* assurance, it is not easy to take place of *RDBMS*.

Distributed transactional databases, therefore, are built, which not only meet the strong consistency requirement of transactional databases, but also support scalability of NoSQL databases. They are considered as one kind of NewSQL databases [18]. *H-Store* [1], *VoltDB* [2], *OceanBase* [3], and *Citus*, [4] design a *Shared-Nothing* model to achieve a distributed database. Specifically, the model takes predefined rules to partition data into multiple (distributed) nodes. Each node covers the functionality modules of *Query*, *Transaction* and *Storage* engines, and coordinates the other ones when and only when executing distributed transactions or distributed queries. Generally, it addresses two key scalability issues, i.e., storage scalability and distributed transaction processing scalability. Nevertheless, it lacks of an effective scheduling mechanism and depends well on pre-defined partitioning rules, which demands great effort from the business developers, i.e., to understand the business. Otherwise, it may lead to frequent distributed transactions, which seriously degrades the overall performance.

Therefore, *Spanner* [5], *TiDB* [6] *CockroachDB* [7] and *FoundationDB* [11] extend the *Shared-Nothing* model by separating compute and storage to achieve flexibility and scalability at the same time, that is to separate the query engine and storage engine. The query engine is state-less and can run in any number of nodes. The storage engine provides transactional storage access with multiple replicas. Meanwhile, instead of relying on the predefined distribution rules, they divide the data into sub-blocks and use a centralized management controller to balance the storage and workload among all storage nodes. Therefore, a compute node can access any storage node with enough flexibility, and schedule data and workload according to optimized goals, e.g., storage or performance. However, it produces worse performance under complex distributed transactions because of introducing too many network interactions (I/Os).

Aurora [8,9], *PolarDB* [10], *Socrates* [13] and *Taurus* [12] then design a *Shared-Storage* model that sacrifices the flexibility and scalability of write to deal with more complex transactional workloads. Specifically, they keep the query engine and transaction management on one compute node and then use multiple storage nodes to store the data and logs. All other compute nodes are read-only nodes, relieving the burden on the write node by separating reads and writes. This architecture has full scalability for storage and read workload, but the single write node has a high probability to bottleneck *DBMS*. However, since all transactions are executed within a single node, their performance is not affected by distributed transactions even with complex transactions.

In summary, different distributed models have been specified and designed for different application scenarios, which may expose high performance for its target application. It is urgent to benchmark these distributed databases to make a fair comparison and benchmarking

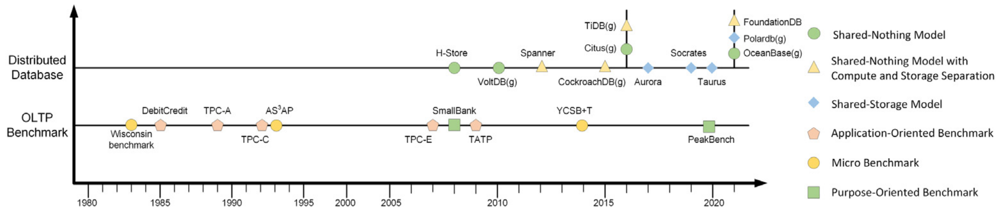


Fig. 1. Development of distributed databases and OLTP benchmarks.

among them, and to expose the choke points of different distributed databases. It will then further promote the development of databases, which is especially useful for database marketing, engineering and targeting sales & innovations.

The commonly used benchmarks for transactional databases (OLTP benchmarks) are presented in Fig. 1, ranging from 1983 to 2020. OLTP benchmarks can be divided into three categories. The first type is $micro$-benchmarks, which consist of simple *CRUD* operations (*Create, Retrieve, Update, Delete*) and do not represent any application or business. These benchmarks, hence, are able to compare the performance of transactional databases with respect to the basic operations. Specifically, university of *Wisconsin* formulates a benchmark for $DBMS$ (also called *Wisconsin* benchmark) [19], which is the first one to define a set of SQL statements to compare the database performance by collecting their total execution time. It, however, has several serious limitations, which are lack of data type supports, no batch update or concurrency control. To deal with these problems, two designers of *Wisconsin* benchmark, i.e., *Turbyfill* and *Bitton*, together with *Orji* introduce *ANSI Sql Standard Scalable and Portable* benchmark, i.e., *AS*^{3}*AP* benchmark [20]. The improvements include not only adding more tables, data types, index types, data distributions and the number of *SQL* statements, but also increasing the volume of data. Furthermore, isolation levels are introduced to evaluate concurrency control. Both *Wisconsin* benchmark and *AS*^{3}*AP* benchmark, however, do not wrap SQL statements into transactions. In addition, when data size continues to increase, the scalability of databases attracts more attention. *YCSB+T* [21], an extension of *YCSB* [22] designed to evaluate $NoSQL$ databases, covers the standard read, write, update, delete and scan operations in *YCSB*, based on which it composes transactions. *YCSB+T* has an additional validation stage to check the consistency of a distributed database to guarantee the correctness of execution.

The second type is the benchmark focusing on evaluating database for different applications. These benchmarks are constructed from the characteristics of the specific applications, which may reflect the business logics. *TATP* [23], proposed by *IBM*, abstracts the operations in a telecommunication business application. It introduces physical resource consumption as one of its metrics. Besides, *Jim Gray* together with *Tandem Company* simulates a *Debit Credit* application of the bank, named as *DebitCredit* [24]. *TPC-A* [25] benchmark by *TPC Council* is built upon *DebitCredit* benchmark, and it is the first time to formalize the evaluation rules including specifying $ACID$ properties, which all database vendors are required to obey. *TPC-A* is rarely used in recent years because it is too simple to satisfy the needs of current applications. Later, *TPC Council* proposes a more complicated benchmark, *TPC-C* [26], which simulates a warehouse and order management application. Most database vendors have participated in *TPC-C* evaluation to demonstrate the performance of their databases, including distributed transaction databases, such as *OceanBase*. *TPC-E* is the most complex transactional benchmark, which simulates the typical behaviors in a stock brokerage company.

The third type of the benchmark is designed for exploring and simulating the specific characteristics of applications, such as *SmallBank* [27] and *PeakBench* [28]. *SmallBank* is designed to evaluate DBMS for the workload with the characteristics of the read–write conflict(called anti-dependency). It can be used to evaluate different serializable protocols under snapshot isolation. It contains simple read and write operations involving a small number of tuples. *PeakBench* is designed to evaluate $DBMS$ for the workload with the characteristics of *sharp dynamics, terrific skewness, high contention*, and *high concurrency*, that is to simulate a second kill application in *Alibaba*.

We arrange the mentioned classic distributed transactional databases and existing transactional benchmarks in Fig. 1 according to the chronological order when the databases/benchmarks are either open sourced in github(labeled as (g)) or published officially. Different shapes and colors represent the different types of databases and benchmarks. It is obvious that most of *OLTP* benchmarks are proposed before 2010, along with the growth of centralized $DBMS$s. The prosperity of distributed databases starts from *H-Store*, the popular distributed database around 2008. Though *PeakBench* is the most recently proposed benchmark transactional databases, it focuses on the performance comparison with the write intensive workload rather than analyzing or benchmarking distributed databases. Besides, the existing surveys cover the fields including $NoSQL$ benchmark [29,30], Big Data system benchmark [31–33], decision support benchmark [32], graph processing system benchmark [34], etc. However, no work conducts a benchmark survey for distributed transaction databases. Therefore, it is imperative to answer whether the existing benchmarks are still applicable to evaluating distributed transactional databases. And if they are not suitable for distributed transactional database evaluation, what are the disadvantages?

The contributions of this paper are: (i) we summarize the representative architectures of distributed transactional databases and expose the potential choke points in their design; (ii) we discuss and analyze the deficiency of the state-of-art benchmarks with respect to these choke points; (iii) we provide a guideline to revise the existing benchmarks or design a new benchmark for benchmarking distributed transactional databases.

2. Architecture and choke points of distributed databases

In this section, we will discuss about the components with respect to the scalability of distributed databases, and then explore the choke points of transaction processing of distributed databases.

2.1. Architecture of distributed databases

Shared-Nothing model and *Shared-Storage* model are two popular distributed database models with various implementations and optimizations. Specifically, the *Shared-Storage* model is proposed for the cloud-oriented databases recently. For *Shared-Nothing*, we introduce two classic kinds of databases, i.e., *Bundle/Separation of Compute and Storage*. For *Shared-Storage*, we mainly introduce *Aurora*, which uses the model in the cloud firstly and then presents the optimization made by others.

2.1.1. Shared-nothing model

Bundle of Compute and Storage As far as we know, *H-Store* [1] is the first distributed *DBMS* with SQL compatibility and $ACID$ support. $VoltDB$ [2] is a business implementation of *H-Store* with the architecture shown in Fig. 2. It divides all data into the main memory of a distributed cluster and uses *K-Safety* mechanism to tolerate errors. Each node in *H-Store* has an *Execution Engine* and *Partition Data*, which are

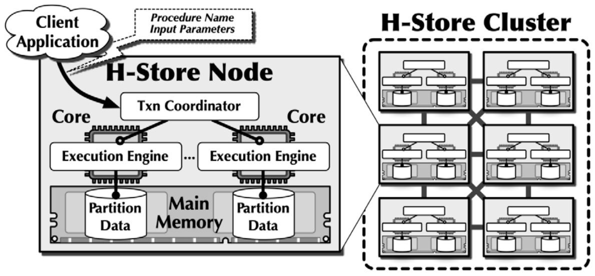


Fig. 2. H-store architecture [1].

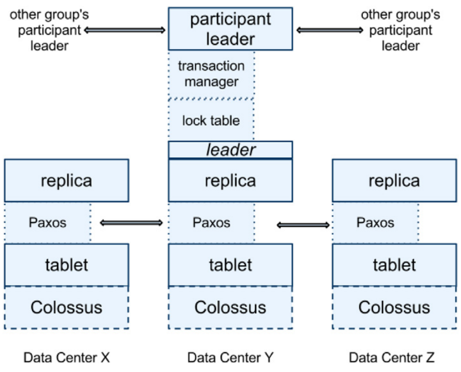


Fig. 3. Spanner architecture [5].

tied together. *H-Store* proposes a novel partitioned-based concurrency control to execute distributed transactions. Specifically, *Txn Coordinator* divides a stored procedure-based transaction into one or more partitions. If and only if all the partitions execute successfully, the transaction can be successfully committed. Otherwise, the transaction fails. Therefore, if the workload on different partitions is unbalanced, the throughput lows down. Additionally, *K-Safety* requires the same replica to be equal, so the existence of replica nodes can be used to relieve read pressure.

Citus [4], as a distributed database plugin for *PostgreSQL*, organizes multiple *PostgreSQLs* into a distributed database cluster. The data is distributed across multiple machines according to the predefined rules, which is synchronized using master–slave replication. *Citus* uses a coordinator node to distribute and coordinate transactions, and launches distributed transactions based on *MVCC+2PL* protocol. However, *Citus* cannot support globally consistent snapshots due to the lack of timestamp synchronization among multiple nodes, which makes it weak in transaction execution.

OceanBase [3] is a commercial distributed database. The data is distributed similarly to *Citus* and achieves consistency among replicas by its self-developed *Paxos* algorithm. *OBProxy* is a stateless connector to clients, which routes *SQL* statements by the proxy table. There can be multiple *OBProxies* for *OceanBase*. When dealing with a distributed transaction, the coordinator is chosen among the nodes involved. Therefore, *OceanBase* no longer distributes and coordinates transactions through a centralized node, which significantly improves the stability and scalability of the distributed *DBMS*.

However, the predefined rules for data partitioning make databases a weak scheduling capability. None of the above databases has adaptive hot data splitting capability. Moreover, the computational resource is tied with data, which lowers the computing elasticity because it is not until the data is moved to the computational node that the computation node works.

Separation of Compute and Storage *Spanner* [5] is published by *Google*, which extends the *Shared-Nothing* model with the separation of compute and storage. Its query engine is defined as the compute layer, where each node is stateless and uncoordinated; its transactional storage engine is the storage layer, where each node is stateful. *Spanner* uses a sharding approach to organize data as multiple *tablets* and applies *Paxos* protocol to synchronize *tablets* among data centers shown in Fig. 3. *Colossus*, a distributed file system, is used to persist data within data centers. *Spanner* takes the *TrueTime* API based on the atomic clock to limit the time deviation of individual data shards around the world, by which it guarantees strict serializability and executes distributed transactions globally. In the compute layer, *Spanner* proposes an MPP-enabled query engine [35] that can execute distributed queries through *TrueTime* API.

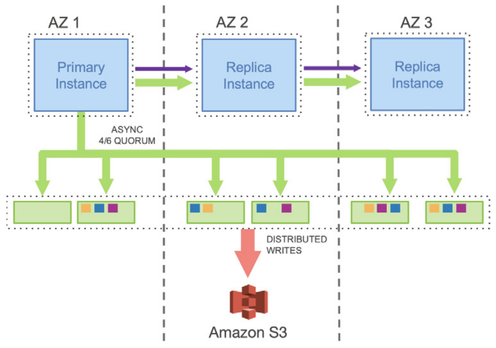


Fig. 4. Aurora architecture [8].

However, *TrueTime* API is not available in most databases. Therefore, *CockroachDB* [7] supports Geo-Partitioning like *Spanner* by leveraging *Hybrid Logical Clocks*. *TiDB* [6] and *FoundationDB* [11] provide the strictly increasing and globally unique timestamps by the central controllers. In addition to the difference in TSO (Timestamp Oracle), *CockroachDB* and *TiDB* also offer better compatibility with traditional databases by providing interfaces to *PostgreSQL/MySQL*. *FoundationDB* further decouples the transaction management from the storage layer, which allows the transaction management to scale independently.

2.1.2. Shared-storage model

Aurora [8,9] is a *Shared-Storage* distributed *DBMS* by separating computing and storage. However, its compute layer contains the query engine and transaction manager, i.e., *Primary Instance*, and the storage layer is only responsible for maintaining multiple replicas of data. Therefore, when extending the compute layer with *Replica Instance*, they can only do read access as there is no transaction manager in the node. For a read node, it is incredibly time-consuming to read all the data from the storage layer, so the read node caches a portion of hot data for subsequent access. However, this cache mechanism is undesirable with multiple writers, because maintaining the cache consistency is more difficult in a distributed system. To ensure that read is from a same global snapshot, *Aurora* requires that the read version from the storage layer should be the same as the version of data in the cache, while read nodes synchronize redo logs from write nodes to update the cache. However, this approach cannot guarantee linearizability. In the storage layer, *Aurora* uses *Quorum* to secure data and separates the log and data (see Fig. 4).

PolarDB [10], *Socrates* [13], and *Taurus* [12] follow the design of *Aurora* and propose some approaches for the storage layer optimization. *PolarDB* designs *PolarFS* based *Remote Direct Memory Access* (*RDMA*) that is a shared distributed file system with *POSIX* interfaces. This design allows fewer changes with the traditional *DBMS* but introduces write amplification. *Socrates* and *Taurus* design the log system and data system to replace the storage layer in *Aurora*, which are optimized for the characteristics of logs and data, respectively.

2.2. Choke points in distributed transaction processing

In this section, we depict the choke points of distributed databases in respect of transactions, queries and task schedulers.

Table 1
Comparison of distributed databases on supporting transaction processing.

Model	Database	Storage		Transaction		Query	Schedule		
		Replica consistency	Global snapshot	Distributed ratio	Concurrency control	Strong Consistency read on replicas	Elasticity computing	Adaptive splitting	Online storage movement
Shared-Nothing	H-Store/VoltDB	K-Safety	T	Sharding Relative	Partition-based	/	T	F	T
	Citus	Master–slave	F	Sharding Relative	MVCC+2PL	F	T	F	T
	OceanBase	Paxos	T	Sharding Relative	MVCC+2PL	F	T	F	T
Shared-Nothing with separating compute and storage	Spanner	Paxos	T	High	MVCC+2PL	T	T	T	T
	TiDB	Raft	T	High	Percolator	T	T	T	T
	CockroachDB	Raft	T	High	Percolator	T	T	T	T
	FoundationDB	K-Safety	T	No	MVCC+OCC	T	T	T	T
Shared-Storage	Aurora	Quorum	T	No	MVCC+2PL	F	Read-only	T	T
	PolarDB	Raft	T	No	MVCC+2PL	T	Read-only	T	T

2.2.1. Transaction

To achieve scalability, data is usually partitioned to multiple database nodes which may lead to distributed transactions. Databases have to face three difficulties when transactions scaled to multiple nodes.

Firstly, for the sake of isolation property, *Concurrency Control* among transactions is a core issue. *MVCC*, *OCC* and *2PL* [36,37] are proposed to ensure the correctness of transaction processing. They are usually used together to guarantee the correctness of concurrent execution. For example, five distributed databases [3–5,9,10] in Table 1 utilize a combination of $MVCC$ and $2PL$. Besides, $Percolator$ is used in *TiDB* and *CockroachDB*, which is a variant of OCC. Due to the complexity of distributed transactions, we need to take into consideration two more factors, i.e., the context of distributed transactions and timestamp management.

- In order to maintain the context of distributed transactions, distributed *2PL* is taken to deal with lock assignment and release among all participators. How to distribute the version information among all participators is vital in *MVCC*, and how to complete verification atomically among all participators is indispensable in *OCC*.
- A global timestamp management is imperative. There are two solutions proposed. One is to provide the strictly increasing unique timestamps by the central timestamp distributor. The other is to maintain a global timestamp service. For example, a *True-Time* API is used in *Spanner* and *Hybrid Logical Clock* is used in *CockroachDB*. When a distributed database is unable to provide a global snapshot, *Read Committed* is the highest isolation level which it can support. Specifically, there is a time gap between the commits from different nodes because each node uses its own timestamp. For this time gap, a transaction may access the inconsistent versions of data from different nodes and repeatable reads are not guaranteed.

Secondly, to ensure correctness of transaction execution and data consistency, commit management is necessary among different nodes. For this purpose, *Two Phase Commit* (*2PC*) and *Three Phase Commit* (*3PC*) are the choices for almost all distributed databases, which usually lead to an obvious increase of *Latencies*.

Lastly, it is widely accepted that the ratio of distributed transactions is highly dependent on data distribution. To reduce distributed transactions, *Tablegroup* is introduced in *Spanner* (*aka*. interleaved table) and *OceanBase*. *Tablegroup* describes locality relationships that exist between multiple tables, namely, putting data that is often accessed together. Due to the lack of such an optimization, the distributed ratio of TiDB and CockroachDB is usually high.

2.2.2. Query

Distributed query processing is popular in a distributed cluster, which meets two main challenges. The first is to provide a global snapshot to guarantee data consistency among different nodes. As in Table 1, all distributed databases provide users with a global snapshot, except *Citus*. But the global snapshot is non-trivial in processing distributed queries [38,39]. The second one is some pivotal features of the distributed databases ought to be taken into account in the phase of query optimization [40]. Then the query optimizer should be adapted to the distributed environment, which shall consider distributed features, e.g., data locality.

A representative processing architecture is to route all client requests to the leader replica, such as in *Citus*. The slave replicas take responsibility of providing availability in case the master node fails. As the explosive increasing of client requests, the salve replicas have been used to handle read requests to alleviate the pressure of the primary replicas and the master node is responsible for write operations. For high performance, all distributed databases in Table 1 prefer to use this strategy. Unfortunately, to achieve the strong consistency among the master and the salves will have a negative impact on the availability based on *CAP* theorem [41]. So most current distributed databases use *Quorum/Paxos/Raft* protocols to guarantee the eventual consistency and allow the existence of soft states, which are designed around the *BASE* philosophy [42]. Based on it, although several distributed databases support a strong consistent read on replicas, as shown in Table 1, they have to wait for synchronizing slaves before responding to clients and hence cause lower performance. Besides, *K-Safety* represents the number of *K* replicas of the data in the distributed database. For example, in *H-Store* and *VoltDB*, duplicating database partitions as members of full functions, that is all the partitions are equal in providing services, i.e., supporting both read and write operations.

2.2.3. Scheduler

Scheduler covers the functionalities of adaptive splitting, elasticity computing and storage movement.

Adaptive Splitting The purpose of the adaptive splitting is to alleviate the pressure from clients on some nodes. When hotspots exist in shards, it takes adaptive sharding to split these shards and then moves some shards to the other nodes. Due to the complexity in maintenance, it is implemented in some databases as shown in Table 1. Specifically, these distributed databases split the hot shards based on the analysis of the historical statistics on workloads and resource consumption.

Elasticity Computing It is designed for managing physical resource adjustment, i.e., increasing or decreasing resources. Usually, when there are not enough resources, it is preferred to flexibly allocate tasks from the busy nodes to the newly-added nodes. From Table 1, all distributed databases present the ability of elasticity computing. However, *Aurora* and *PolarDB* scale read-only nodes because they have

Table 2
Comparison of benchmarks in benchmarking distributed transactional databases.

	Choke points	Transaction			Query		Scheduler		
		Contention	Distributed transaction	Consistency testing	Distributed query	Read–write separation	Adaptive splitting	Computing elasticity	Storage movement
Micro benchmark	YCSB+T	Yes	doTransaction-ReadModifyWrite	Yes	doTransactionScan, doTransactionRead	Yes	Yes	/	/
Application-oriented benchmark	DebitCredit	/	Yes	/	/	/	/	/	/
	TPC-A	/	Yes	Yes	/	/	/	/	/
	TPC-C	/	NewOrder	Yes	/	Yes	/	/	/
	TATP	Yes	/	/	/	Yes	Yes	/	/
	TPC-E	/	Market-feed, Trade-cleanup, Trade-result, Trade-update	Yes	Trade-cleanup	Yes	/	/	/
Purpose-oriented benchmark	SmallBank	Yes	Amalgamate	/	/	Yes	Yes	/	/
	PeakBench	Yes	Submit order, Pay order, Cancel order, Overdue order	/	Q2-Q5	Yes	Yes	Yes	/

a single node for write. There is a relatively lower scaling cost in *Shared-Nothing* architecture with separating computation and storage than the one without resource separation, because the latter has to scale both of the resources, i.e., compute and storage, and meets more complex maintenance.

Storage Movement Storage movement aims to distribute the shards evenly across nodes, alleviating the storage pressure and avoiding disk explosion on a single node. To enable an even distribution, it moves shards until it reaches an even number of shards across nodes. For distributed databases in Table 1, all of them realize this functionality but through different strategies. For example, a shard rebalancer is provided in *Citus*. The second responsibility of storage movement is to put the shards frequently accessed together from clients into the same nodes. The rebalancer moves these shards into one node by analyzing historical statistics. This feature has been extensively studied [43,44]. Due to its high implementation cost, it is not supported by all distributed databases.

2.3. Benchmark overview

Many benchmarks have been designed to evaluate databases [27, 28,45–48]. We classify them into *Micro Benchmark*, *Application-oriented Benchmark*, and *Purpose-Oriented Benchmark*. A *Micro Benchmark* is either a program or routine to measure and test the performance of a single component or task in a (huge) system or program [49]. It cares more about the performance of a sub-component. Both *Application-oriented Benchmarks* and *Purpose-Oriented Benchmarks* are macro benchmarks [50], which expect to simulate the representative workloads (business logic) of an application and are used to evaluate the overall performance of a system. *Purpose-Oriented Benchmark* is a special case of *Application-oriented Benchmark*, which focuses more on a specific characteristics of applications, such as terrific contention and skewness in PeakBench for SecKill application and SI isolation level testing in SmallBank.

Then there is a question that whether these benchmarks are still applicable to distributed databases. Considering data placement rules, e.g., $Hash$ or *Range*, we suppose that tables with dependencies are divided according to primary keys of the referenced table, i.e., partition keys. Based on this, a distributed transaction is defined if a transaction has different *partition IDs* on its modification operations, i.e., *insert, deletes and updates*; a distributed query accesses data from different *partition IDs*. Adaptive Splitting may be triggered and storage movement may happen for balancing workload. Supporting for both scale factors in data and workloads has a requirement for elasticity computing.

We try to study these benchmarks from data schema, workload, performance metrics and execution rules to explore the ability to evaluate distributed transactional databases with respect to the chock points in the layers of *storage, query* and *schedule* and fill the results in Table 2. Specifically, "/" means this benchmark misses the corresponding test requirement; "Yes" means that this benchmark meet the requirement. Two items, i.e., Distributed Transaction and Distributed Query, demonstrate the specific transactions which cover the requirement.

3. Micro benchmarks for transactional databases

Micro Benchmarks are proposed for proving the success of a design or an algorithm in some database components [51–53]. Since they are not designed to test databases in a whole, we only mention two popularly used micro benchmarks here. One is *AS*^{3}*AP* for the early databases [19, 20] and the other is *YCSB+T* for the *NoSQL* databases [21]. For traditional databases, usually the single-node one, it is tough to scale due to the lack of distributed consistency algorithms and distributed concurrency control protocols. So the benchmark is not suitable for testing the critical features of distributed transactional databases. For the benchmark evolving from *NoSQL* databases, it provides transactional operations and supports scaling out, but it does not take computing elasticity and storage movement into account, which is preferred for load balance in distributed databases.

3.1. AS^3AP

AS^3AP [20] is designed to make up for the shortcomings of *Wisconsin* Benchmark, by covering multi-user test, more data types, data distributions, etc.

3.1.1. Schema and data generation

AS^3AP involves five tables, i.e., *Uniques*, *Hundred*, *Tenpct*, *Tiny* and *Updates*, covering eleven common data types. *Tiny* is used to measure overhead during benchmarking, while the other four are the common data tables. Only *Hundred* and *Updates* have reference relationship. These four tables can scale from 1 MB to 100 GB.

3.1.2. Workload

AS^3AP test has two modules, i.e., *single-user test* and *multi-user test*. *Single-user test* covers running context preparation and user queries. Context preparation includes *load, backup, building indices*, etc., while user queries include *retrievals, single-tuple updates, and bulk updates*. During test, the system will be penalized if it does not use parallel distributed algorithms as the data volume increases. Therefore, it is suitable to evaluate the ability of query processing in distributed databases to some extent. AS^3AP takes isolation into account when defining *multi-user test*, but it still sends operations as queries rather than wrapping *CRUD* operations into transactions. Therefore, it is not suitable to test the atomicity of transactions, which is especially important for distributed transactional databases because it involves core modules for parallel transaction processing, e.g., *Distributed Transaction Manager* or *Distributed Commit Protocol*. So we do not list it in the Table 2.

3.1.3. Evaluation and performance metrics

There are two key evaluation metrics proposed by AS^3AP, i.e., *equivalent database ratio* and *cost per megabyte*. Both of them are based on *equivalent database size*, i.e., the maximum size of database generated by performing the designated set of single-user and multi-user tests under 12 h. *Equivalent database ratio* is designed to compare two systems under test by a division operation between their *equivalent database sizes*. *Cost per megabyte* is the total cost (execution time) of the database divided by the *equivalent database size*.

3.2. YCSB+T

YCSB+T [21] is an extension of *YCSB* [22], which aims to evaluate the transactional capability of *NoSQL* databases. Since there is no a widely accepted declarative language like SQL on the interface of *NoSQL*, *YCSB+T* and *YCSB* just have simple *read/write* operations.

3.2.1. Schema and data generation

NoSQL database defines a table structure instead of relational models to support various types of NoSQL applications, such as Key–Value, Document, etc. *YCSB+T* has a table with two columns. One column is the primary key and the other is a field called *money* with default value \$1000. *Money* in *YCSB+T* simply simulates a closed economy business, in which money does not enter or exit the system during the evaluation period.

3.2.2. Workload

YCSB+T provides six types of transactions by wrapping *CRUD* operations. They are *insert, random-scan, range-scan, update, delete* and *read–modify–write*. Specifically, *insert, update* and *delete* transactions cover only a single statement accessing one row, and all of which are local transactions. *Random-scan* and *range-scan* transactions require to get a consistency snapshot of the distributed storage, which are distributed queries in a high probability. *Read–modify–write* transaction reads and writes two rows each time, which may be distributed transactions. *YCSB+T* follows the design of *YCSB* in setting the data access distribution of *read/write* by a configuration file, which may create hotspots in writing when a large number of transaction happens.

3.2.3. Evaluation and performance metrics

YCSB+T and *YCSB* both use ops/s (the number of operations that can be performed per second) as a metric for performance. Meanwhile, with the closed economy scenario, *YCSB+T* can determine whether the consistency constraint is satisfied by counting the total amount of all accounts before and after business execution. If it cannot guarantee strict consistency, the *exception score* of the database is measured by the deviation value before and after business execution divided by the data volume.

4. Application-oriented benchmarks for transactional databases

In this section, we introduce five popularly used classic benchmarks designed for specific applications, i.e., *DebitCredit* [24], *TPC-A* [25], *TATP*, [23], *TPC-C*, [26] and *TPC-E* [54]. We declare their support for benchmarking distributed transaction databases.

4.1. Debitcredit and TPC-A

DebitCredit, first introduced in 1985 by *Jim Gray*, is an online transaction processing benchmark, simulating a virtual multi-branch bank transaction system. Compared to *Wisconsin* benchmark [55], *DebitCredit* proposes a benchmark much closer to the real database system with various performance metrics. *TPC Council* adds some new features and standardizes this benchmark, formalized as *TPC-A*.

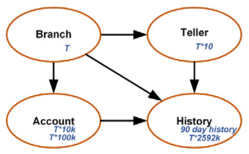


Fig. 5. Data model of *DebitCredit* and *TPC-A*.

4.1.1. Schema and data generation

The database model for *DebitCredit* is quite simple, which contains four tables as shown in Fig. 5. If the performance goal of *TPS* is T, *DebitCredit* specifies that the database should have at least T records in *Branch*, $T * 10$ records in *Teller*, $T * 10k$ records in *Account*, and a 90-day history record in *History*. *TPC-A* has different scale ratio compared to *DebitCredit*. The number of rows of *Account* and *History* in *TPC-A* are $T * 100k$ and $T * 2592k$, respectively.

DebitCredit does not demonstrate the partition strategy in detail, while *TPC-A* stipulates the specific partitioning rules. For *TPC-A*, it is horizontally partitionable, that is to shard data to different nodes based on primary key in *Branch*. Therefore, as the volume of data increases, it has a good property to be linear scale-up. Data generation in *DebitCredit (TPC-A)* is based on a specific distribution. They pay more attention to the generation of primary keys, while rarely delve into any details about the non-key columns.

4.1.2. Workload

DebitCredit (TPC-A) has only one transaction with simple write operations. Currently all isolation levels of most databases can avoid write conflicts, including distributed databases, so *DebitCredit (TPC-A)* can be used to evaluate the ability of distributed databases in all isolation levels. Specifically, this transaction portrays that a customer makes a withdrawal or deposit at a bank by updating his *Account*, while updating *Teller* and *Branch* simultaneously to maintain data consistency [56]. An account is randomly selected from a remote branch with a probability of 15%, which causes distributed transactions spanning across two nodes. Then, it keeps appending the modifications to table *History* sequentially. Due to the fact that no query operation involved in this transaction, distributed queries do not exist.

4.1.3. Evaluation and metrics

DebitCredit is not designed for testing $ACID$ properties, while *TPC-A* requires the database to meet $ACID$ properties and puts forward a series of tests which must be launched by vendors. Concretely, atomic tests verify that for any randomly selected account, the records in the related table will change synchronously (or remain the same) after committing (or aborting). It requires database to conform to additional three predefined conditions in consistency tests. Moreover, isolation and durability tests are also strictly defined in *TPC-A*. Hence, *TPC-A* can be used to evaluate whether the distributed databases meet *ACID* properties.

The primary metrics in *DebitCredit (TPC-A)* include *Elapsed time*, *TPS* and *Cost*. *Elapsed time* indicates the time duration to do one standard batch of transactions on a database. *TPS* reports maximum transactions per second a database can achieve before system is saturated. Considering the cost of maintaining hardware/software components utilized, *Cost (price/TPS)* is provided. Lower *cost* means that adding extra resources can obtain larger *TPS*.

Unfortunately, due to the fact that *TPS* is strictly bound with data size, it fails in making data hotspots with respect to workloads. So the processing ability with intensive contention situation and scalability of transaction processing in distributed databases cannot be well evaluated. In addition, scheduling evaluation has not been mentioned by *DebitCredit (TPC-A)* due to its low throughput.

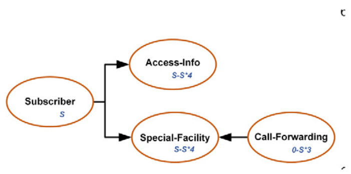


Fig. 6. Data model of *TATP*.

4.2. TATP

TATP benchmark simulates a typical Home Location Register (HLR) scenario used by a mobile carrier. The special application scenario makes it a perfect example of a demanding high-throughput environment.

4.2.1. Schema and data generation

TATP consists of four tables as shown in Fig. 6. Specifically, the number of rows of other tables depend on the number of rows of table *Subscriber*, i.e., S, and scale out according to the defined rules. For instance, each subscriber owns one to four records in table *Access-Info*. The dependencies indicate that the schema provides a convenient way to scale to distributed databases by partitioning on *Subscriber ID*.

The primary keys are generated sequentially, while the non-key columns are randomly generated except *sub_nbr*. *Sub_nbr* is a string-typed column generated from primary key in table *Subscriber*, and it is also used to retrieve the related primary key in certain transactions.

4.2.2. Workload

There are seven transactions in *TATP*, including *Get-Subscriber-Data*, *Get-New-Destination*, *Get-Access-Data*, *Update-Subscriber-Data*, *Update-Location*, *Insert-Call-Forwarding* and *Delete-Call-Forwarding*. The first three transactions require *Read Committed* isolation level, while the last four transactions need *Repeatable Read* isolation level. Therefore, *TATP* is not suitable for distributed databases that cannot provide isolation level of *Repeatable Read* or the one stricter than *Repeatable Read*, which requires a global snapshot, e.g., *Citus* [4].

Read-only transaction exists in *TATP*, i.e., *Get-Subscriber-Data*, *Get-New-Destination*, *Get-Access-Data*. For example, *Get-Subscriber-Data* retrieves one row from table *Subscriber*. Applications with read–write separation architecture can take these transactions for performance evaluation. What is more, write transactions are all single-point operations and no distributed transaction processing exists. For all transactions, *Subscriber ID* is generated randomly using non-uniform distribution by default. This generation mechanism may create some hotspots on data when scaling to a distributed environment under a large number of transactions. As a consequence, intensive contentions are likely to occur on nodes with heavy loads. Besides, it can be used to evaluate the feature of adaptive sharding based on the distribution of primary keys.

4.2.3. Evaluation and performance metrics

TATP collects two types of results when benchmarking, i.e., *MQTh* (Mean Qualified Throughput) and *transaction response time distributions*. *MQTh* is the number of successful transactions per time unit. The *response time* is measured for each individual transaction and reported for each type of transaction.

4.3. TPC-C

TPC-C, introduced in 1992, is an on-line transaction processing benchmark, simulating a warehouse-centric order processing application, which is one of the most popular benchmarks and widely used to demonstrate the performance of databases

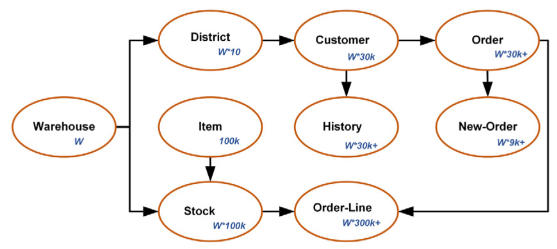


Fig. 7. Data model of *TPC-C*.

4.3.1. Schema and data generation

TPC-C consists nine tables and the reference relationships among them are shown in Fig. 7. The number of rows in each table, except *Item*, are dependent on the number of *Warehouse*, i.e., *W*, and scale out according to a specific rule. For example, each warehouse provides the services for 10 districts. All tables except *Item* can partition their data according to *Warehouse ID*. Therefore, it has the good property to be scaled (increasing the number of W) to distributed nodes by splitting *Warehouse ID* when increasing the volume of *TPC-C* workloads.

Data generation in *TPC-C* is based on a specific distribution. The primary keys are generated sequentially, while the non-key columns are randomly generated except *Customer Last Name* ($CLast$). *CLast* is generated according to a specific rule for subsequent queries. Based on the running statistics, the data size of a warehouse and its relative table is 70 MB without compression, which needs to increase the number of warehouses in order to test current advanced database systems.

4.3.2. Workload

There are five transactions in *TPC-C*, including *New-Order*, *Payment*, *Order-Status*, *Delivery* and *Stock-Level*. All transactions except *Stock-Level* require to run in *Repeatable Read* isolation level, while *Stock-Level* need meet *Read Committed* isolation level. Based on this, databases which do not support a global timestamp cannot pass the isolation level test, such as *Citus* [4]. There are four transactions, i.e., *NewOrder*, *Payment*, *Stock-Level*, *Order-Status*, which are related to distributed processing.

New-Order describes that a customer creates an order and inserts the order and item information into 4 involved tables. It represents a read–write transaction with a high frequency of execution. When the item is supplied from a remote warehouse (1%), it is likely to cause a distributed transaction. Unfortunately, it does not take the number of nodes spanning across database into consideration, namely, unable to valuate $2PC$, $3PC$ and their optimized versions. Besides, all but one tables are dependent on *Warehouse*, so $tablegroup$ can be created to decrease the distributed transaction ratio.

Payment portrays that a customer pays for the order by updating his *balance*, *sale statistics* in *District* and *Warehouse*. Distributed queries may exist in *TPC-C*, because they involve non-key column read. However, if the row number of customers under a warehouse can be put into a partition (which is normal), distributed queries disappear. Besides, a customer is randomly selected from remote warehouses in 15% probability, which causes distributed transactions spanning across 2 nodes.

Stock-Level is a read-heavy transaction depicting the total number of recently sold items whose stock level is below a certain threshold, while *Order-Status* is a read-only transaction which queries the status of a customer's last order. Both transactions are suitable for evaluating databases which support a read–write separation architecture. Besides, although they are read-heavy transactions, both of them access a single node. So distributed query still does not exist in these two transactions.

4.3.3. Evaluation and performance metrics

TPC-C requires the database to meet *ACID* properties and specifies a series of tests which must be performed by database vendors just like

TPC-A. Therefore, it can be used to evaluate whether the distributed databases meet $ACID$ properties.

Think time and *keying time* are used to simulate the trading in real-life operations. Keying time means the time that a terminal keys the keyboard, and think time is the time to decide which product to choose. Both of them allow a certain interval of execution between transactions. Therefore, they decide the upper limitation of throughput defined by *TPC-C*, i.e., *12.86 tpmC* per warehouse. Unfortunately, it cannot create any hotspots on data, which cannot be used to evaluate the ability in intensive contention processing and scalability of transaction processing of distributed databases. Besides, the scheduling which includes automatic data splitting and storage movement, is not involved in *TPC-C* due to low throughput.

The primary metrics in *TPC-C* include *tpmC*, *price/tpmC*, *availability date* and *watts/KtpmC*. *TpmC* reports the number of orders processed per minute. In order to consider the cost of maintaining the hardware/software components on which the database is running, *price per tpmC* is calculated, i.e., *price/tmpC*, which is able to evaluate the scalability of the distributed databases. That is, when adding a new node, a smaller price/tpmC means the better scalability of database. *Availability date* defines how much time it takes to reach a stable state. Finally, *watts/KtpmC* exposes the electricity consumption per $tpmC$.

4.4. TPC-E

TPC-E, introduced in 2007, is an online transaction processing benchmark, simulating the activity of a stock brokerage firm. The complex business semantics make it difficult be applied to evaluate database performance.

4.4.1. Schema and data generation

TPC-E consists of 33 tables, which can be organized into four types, i.e. *Customer Tables, Broker Tables, Market Tables* and *Dimension Tables*. The number of rows in each table is dependent on the number of customers, and scales out according to a predefined scale factor. For example, the number of trades equals to $17280 \times customers$.

The schema of *TPC-E* provides a way to scale to distributed databases. Tables can partition data by *Customer ID*, except *Industry, Sector, Status_Type, Broker, Trade_Type, Charge, New_Item* and *Taxrate*, while most of these tables are small. Therefore, it has a good property to be scaled to distributed nodes by splitting on *Customer ID* while increasing the number of customers. On the basis of statistics, when the number of rows in table $Customer$ is 5000, its relative table is more than 50 GB without compression [57], which is much bigger than data generated by *TPC-C*. Data generation in *TPC-E* is based on a pseudo-real data distribution [58], which demonstrates a higher degree of skewness than a totally random way.

4.4.2. Workload

TPC-E has 12 types of transactions. Though some of these transactions are likely to cause distributed transactions or queries, e.g., *Market-Feed*, it is hardly to quantify the ability of distributed transactions processing of databases, for it has no strict definition (control) to distributed transactions in both ratio and distribution scale on nodes.

Market-Feed first updates the prices for securities and then updates trades associated with pending limit orders. These trades in table *Trade* may be stored in different nodes, which is likely to cause a distributed transaction. *Customer-Position* is a read-only transaction, which is suitable for evaluating the processing ability of databases supporting a read write separation architecture. However, *Customer-Position* only accesses a single node, distributed queries do not exist. *Trade-Cleanup* is used to cancel any pending or submitted trades from a database. This transaction may update many rows in table *Trade*, which may locate in different nodes due to different C_ID, i.e., distributed transactions occurs. Besides, in order to detect submitted trades, it executes a distributed query, e.g., *select T_ID from TRADE where T_ID ≥ trade_id and T_ST_ID = st_submitted_id*, which may read trades related to different C_ID. *Trade-Update* is designed to emulate the process of making mirror corrections or updates to a set of trades. In this transaction, $Trade\_ID$ is updated non-uniformly, so hotspots may be caused when a large number of transactions happen. Based on this, intensive contention and adaptive splitting will happen.

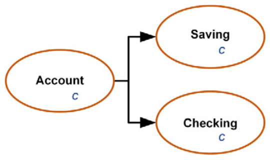


Fig. 8. Data model of SmallBank.

4.4.3. Evaluation and performance metrics

TPC-E requires the database to meet *ACID* properties and launches the same testings as in *TPC-A* and *TPC-C*. The primary metrics in *TPC-E* include $tpsE$, $price/tpsE$, *availability date* and $watts/tpsE$. $tpsE$ reports the number of transactions processed per second. $price/tpsE$ is calculated to measure the cost of maintaining the hardware/software components while running the database, which is able to evaluate the scalability of the distributed databases. When adding a new node, a smaller $price/tpsE$ means the better scalability of a database. *Availability date* and *watts/tpsE* are also the same as *TPC-C*.

5. Purpose-oriented benchmarks for transactional databases

Some benchmarks are designed for a specific purpose with respect to applications. Here, two benchmarks are studied, i.e., *Smallbank* [27] and *Peakbench,zhang2020benchmarking*. *Smallbank* verifies the performance of different serializable protocols under a snapshot isolation, and *PeakBench* provides a way to generate the expected contentions among transactions. Besides the normal business logic defined by *Application-Oriented Benchmarks*, it emphasizes the specific characteristics inside the applications (workload), e.g., high concurrency, high contention or high dynamics in PeakBench [28].

5.1. SmallBank

SmallBank, introduced in 2008, is a benchmark that evaluates different serializable protocols under a snapshot isolation (SI). *SmallBank* is based on the example of the anomaly under SI, and abstracts some functionalities simulating a small banking system, where each customer has a pair of accounts, one for savings and the other for checking.

5.1.1. Schema and data generation

SmallBank consists of three tables and the reference relationships among these tables are shown in Fig. 8. The schema of *SmallBank* provides a way to scale to a distributed database, that is to shard data according to $Customer\_ID$ for all these three tables. Besides, the number of rows of both *Saving* and *Checking* are equal to the one of *Account*, i.e., C in Fig. 8. Therefore, by splitting on $Customer\_ID$, increasing customers (data) can be well distributed on nodes (clusters).

5.1.2. Workload

There are 5 types of transactions in *SmallBank*, including *Balance*, *Deposit-Checking*, *Transact-Saving*, *Amalgamate* and *Write-Check*. All transactions except *Amalgamate* are related to one customer, which then cause local transactions.

Amalgamate transfers the total balance in checking and savings of one customer to the balance of another customer, which may cause distributed transactions spanning at most two nodes.

Further, since customers in *Amalgamate* are generated randomly, the ratio of distributed transactions is uncontrollable. Additionally, in

SmallBank, 90% of transactions involve one customer who is selected from a fixed portion of all customers, which can lead to hotspots. Therefore, the processing ability of contention and automatic data splitting or data moving may be evaluated.

5.1.3. Evaluation and performance metrics

To evaluate the ability of different concurrency control protocols for serializability under SI, the metric in *SmallBank*, i.e., *throughput relative to SI*, is the ratio between the *throughput* under serializability and the one under SI. The larger *throughput relative to SI*, the better the performance of concurrency control protocols achieving serializability. However, the most popular distributed databases barely implement serializable under SI, so *SmallBank* cannot evaluate concurrency control protocols in a distributed database at present.

5.2. PeakBench

With critical transaction processing requirements of new applications, innovative database technologies are designed for dealing with highly intensive transaction workloads with characteristics of *sharp dynamics*, *terrific skewness*, *high contention*, and *high concurrency*, e.g., second kill in *Alibaba* ($SecKill$). *PeakBench* [28] defines a package of workloads simulating intensive transactional processing requirements by designing a fine control in contention generation.

5.2.1. Schema and data generation

PeakBench has eight tables which are designed for testing different transaction processing architecture. For a normal daily transaction process, it involves five basic tables, i.e., *Item*, *Customer*, *Supplier*, *Orders* and *OrderItem*; for *SecKill* with optimized queue structures, *SecKillPlan* and *SecKillPay* are involved to speed up the processing; for a commonly used read–write separation architecture, *R_Item* is created for read-only operations.

5.2.2. Workload

In *SecKill*, the activities are divided into two phases. Before the critical moment in *SecKill* scenario, i.e., the start time of the $kill$ activity, there are 4 types of read transactions and one update-only transaction. Read transactions dominate user activities while the update transaction is applied only in read–write separation architecture for data synchronizing between *RDB* (Read Database) and *WDB* (Write Database). After the critical moment in $SecKill$, there are 5 types of $write$ transactions. For queries, it has not an obvious splitting dimension to guarantee items for Q_{2-5} all located in one node. If data are distributed into clusters, distributed queries are inevitable. For writes, since it is difficult to find the co-partitioning columns among tables, transactions of type PO, CO, SuO and OO can all be distributed ones on the randomly generated parameters, among which SuO is high intensive during *SecKill* and causes hotspots in high probability (for killing hot items). *PeakBench* is the first work providing *Contention Ratio* and *Contention Intensity* to define contention status. Since there are hot items, *PeakBench* can test the scheduling ability of databases, e.g., *adaptive splitting* or *data moving*.

When the size of workloads increases, more nodes may be added to support transaction processing, if it is a distributed database. *PeakBench* supports dynamic adjustment of the quantity of workload and it can measure the ability of database *elasticity*.

5.2.3. Evaluation and performance metrics

The main contribution of $PeakBench$ is to provide a fine granularity control on contention generation, which cannot be well controlled by existing work. It does not mention the details of data scaling and do not declare the way to test *ACID* properties of databases. Besides the traditional metrics, *PeakBench* defines a new metric to evaluate the performance stability of database, *Sys_Stability*, when meeting the dynamic or intensive workload.

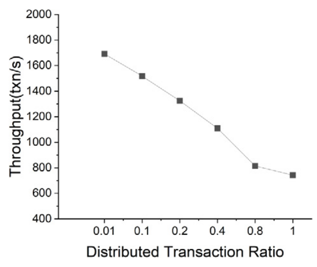


Fig. 9. Different distributed transaction ratio of TPC-C on OceanBase.

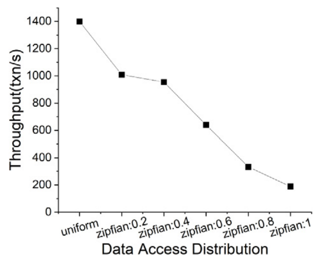


Fig. 10. Different data access distribution of TPC-C on OceanBase.

6. Experiment

We explore the performance under different distributed transaction ratios and contentions, which attract the most efforts in transactional database design. We expect to show database performance is sensitive to these factors, benchmarking on which is urgent and necessary.

Environment. We deploy a distributed database OceanBase (v3.1.1) on a 10-node cluster with 9 OBSevers and 1 OBproxy. Each OBSever is deployed in a server node with one 8-Intel-Cascadelake 6248R @ 3.0 GHz CPU and 32 GB of RAM. OBproxy is deployed with one 16-Intel-Cascadelake 6248R @ 3.0 GHz CPU and 16 GB of RAM. Client is deployed in the same server as OBproxy. All servers are connected using Gigabit Ethernet.

Workloads. We extend *TPC-C* [26] by controlling distributed transaction ratios and the intensity of contentions. In the origin *TPC-C*, *NewOrder* updates 5–15 items in table Stock, covering 1% distributed updates. We expose and parameterize the distributed transaction ratio to evaluate database performance under different distributed transaction ratios. Previously warehouses have a uniform access distributions, we extend the data access distribution in choosing *WarehouseID* to generate different contentions.

6.1. Different distributed transaction ratio of TPC-C

In Fig. 9, we show the throughput of *NewOrder* on OceanBase by adjusting distributed transaction ratios. The throughput drops drastically by 21.6% from 1% to 20%. It demonstrates that different distributed transaction ratios have great influence on performance, which should be taken seriously in benchmarking distributed transactional databases.

6.2. Different data access distribution of TPC-C

In Fig. 10, we show the throughput of *TPC-C* on OceanBase by changing data access distributions of $WarehouseID$. The access distribution covers uniform and Zipfian with parameter s set to 0.2, 0.4, 0.6, 0.8 and 1. The throughput drops obviously by 86.6% from $uniform$ to $zipfian$ with $s = 1$. It shows that contentions affect performance easily, which should also be taken into consideration in benchmarking.

7. Support tools for benchmarking

There have been a set of tools developed for benchmarking so as to simplify the evaluation. *OLTP-Bench* [59] is a popularly used *DBMS* benchmarking testbed, involving 15 kinds of benchmarks. *Smallbank*, *TATP* and *TPC-C* which we have discussed above are all included in *OLTP-Bench*. It supports to connect to multiple *DBMSs* through a component called *SQL-Dialect Manager*. Through a configuration file, it is easy for users to customize their evaluation requirements, such as controlling request rates or *data volume*. Unfortunately, the implementation of some benchmarks does not exactly follow the original definitions. Let us take *TPC-C* for example. *Delivery* transaction intends to be executed in a deferred mode through a queuing mechanism, but it is executed interactively as *New-Order* transactions in *OLTP-Bench*. Besides, the standard metric, i.e., $tpmC$, is not involved in its performance report. Most importantly, it does not provide *ACID* verification during transaction executions. *Benchmarksql* [60] is developed specifically for *TPC-C*. It takes $tmpC$ in its performance report, but still implements *Delivery* transaction in an iterative way. Compared to two tools above, some tools is less common. *tpcc-mysql* [61], the implementation of *TPC-C* is specifically used to evaluate databases supporting mysql protocol. Besides, *tpce-mysql* [62], *DBT-5* [63] and *EGen* [64] are the implementation of *TPC-E*. Specifically, *EGen* is implemented by *TPC Council*, but it is too complicated to run. In addition, *Benchmark Factory* supports the implementation of *TPC-C*, *TPC-E* and AS^3AP, but it is not opensourced. *PeakBench* implements its benchmark and is opensourced in github [28], but it only focuses on contention simulation.

8. Conclusion

In this paper we provide a comprehensive review of several transactional benchmarks about their applicability on evaluating distributed transactional databases. We first introduce two popular distributed database architectures, and summarize choke points in these databases which attract great effort in database design. After that, the paper reviews the classic transactional benchmarks. For each benchmark, we analyze whether it can evaluate the choke points of the distributed transactional databases.

Among the classic benchmarks, YCSB+T, TATP, SmallBank and PeakBench are good choices if we want to measure the performance of the distributed transactional databases under contentions. Although the above mentioned benchmarks, except TATP, all involve distributed transactions, none of them can control the ratio of distributed transactions and the number of spanning nodes, which have great influence on database performance [65,66]. Considering the influence of distributed queries, users can use YCSB+T, TPC-E and PeakBench. As for computing elasticity, PeakBench is a good choice. Finally, none of the benchmarks take storage movement into consideration, which is preferred for load balance processing in distributed databases. To sum up, on one hand, existing benchmarks can be altered for evaluating some aspects of the choke points. On the other hand, considering the evolution and maturity of distributed transactional databases in the future, a new benchmark exploring all the choke points together with an easy-use support tool is imperative for promoting both development and fair benchmarking.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is partially supported by National Science Foundation of China (No. 62072179), ECNU-OceanBase Joint Lab of Distributed Database System and 2020 the Key Software Adaptation and Verification Project (Database). Rong Zhang is the corresponding author.

References

[1] R. Kallman, H. Kimura, J. Natkins, A. Pavlo, A. Rasin, S. Zdonik, E.P. Jones, S. Madden, M. Stonebraker, Y. Zhang, J. Hugg, D. J. Abadi, H-store: a high-performance, distributed main memory transaction processing system, Proc. VLDB Endow. 1 (2) (2008) 1496–1499.

[2] M. Stonebraker, A. Weisberg, The VoltDB main memory DBMS, IEEE Data Eng. Bull. 36 (2) (2013) 21–27.

[3] OceanBase, https://www.oceanbase.com/docs/.

[4] U. Cubukcu, O. Erdogan, S. Pathak, S. Sannakkayala, M. Slot, Citus: Distributed PostgreSQL for data-intensive applications, in: Proceedings Of The 2021 International Conference On Management Of Data, 2021, pp. 2490–2502.

[5] J.C. Corbett, J. Dean, M. Epstein, A. Fikes, C. Frost, J.J. Furman, S. Ghemawat, A. Gubarev, C. Heiser, P. Hochschild, W. Hsieh, S. Kanthak, E. Kogan, H. Li, A. Lloyd, S. Melnik, D. Mwaura, D. Nagle, S. Quinlan, R. Rao, L. Rolig, Y. Saito, M. Szymaniak, C. Taylor, R. Wang, D. Woodford, Spanner: Google's globally distributed database, ACM Trans. Comput. Syst. (TOCS) 31 (3) (2013) 1–22.

[6] D. Huang, Q. Liu, Q. Cui, Z. Fang, X. Ma, F. Xu, L. Shen, L. Tang, Y. Zhou, M. Huang, W. Wei, C. Liu, J. Zhang, J. Li, X. Wu, L. Song, R. Sun, S. Yu, L. Zhao, N. Cameron, L. Pei, X. Tang, TiDB: a Raft-based HTAP database, Proc. VLDB Endow. 13 (12) (2020) 3072–3084.

[7] R. Taft, I. Sharif, A. Matei, N. VanBenschoten, J. Lewis, T. Grieger, K. Niemi, A. Woods, A. Birzin, R. Poss, P. Bardea, A. Ranade, B. Darnell, B. Gruneir, J. Jaffray, L. Zhang, P. Mattis, Cockroachdb: The resilient geo-distributed sql database, in: Proceedings Of The 2020 ACM SIGMOD International Conference On Management Of Data, 2020, pp. 1493–1509.

[8] A. Verbitski, A. Gupta, D. Saha, M. Brahmadesam, K. Gupta, R. Mittal, S. Krishnamurthy, S. Maurice, T. Kharatishvili, X. Bao, Amazon aurora: Design considerations for high throughput cloud-native relational databases, in: Proceedings Of The 2017 ACM International Conference On Management Of Data, 2017, pp. 1041–1052.

[9] A. Verbitski, A. Gupta, D. Saha, J. Corey, K. Gupta, M. Brahmadesam, R. Mittal, S. Krishnamurthy, S. Maurice, T. Kharatishvilli, X. Bao, Amazon aurora: On avoiding distributed consensus for i/os, commits, and membership changes, in: Proceedings Of The 2018 International Conference On Management Of Data, 2018, pp. 789–796.

[10] W. Cao, Z. Liu, P. Wang, S. Chen, C. Zhu, S. Zheng, Y. Wang, G. Ma, PolarFS: an ultra-low latency and failure resilient distributed file system for shared storage cloud database, Proc. VLDB Endow. 11 (12) (2018) 1849–1862.

[11] J. Zhou, M. Xu, A. Shraer, B. Namasivayam, A. Miller, E. Tschannen, S. Atherton, A.J. Beamon, R. Sears, J. Leach, D. Rosenthal, X. Dong, W. Wilson, B. Collins, D. Scherer, A. Grieser, Y. Liu, A. Moore, B. Muppana, X. Su, V. Yadav, Foundationdb: A distributed unbundled transactional key value store, in: Proceedings Of The 2021 International Conference On Management Of Data, 2021, pp. 2653–2666.

[12] A. Depoutovitch, C. Chen, J. Chen, P. Larson, S. Lin, J. Ng, W. Cui, Q. Liu, W. Huang, Y. Xiao, Y. He, Taurus database: How to be fast, available, and frugal in the cloud, in: Proceedings Of The 2020 ACM SIGMOD International Conference On Management Of Data, 2020, pp. 1463–1478.

[13] P. Antonopoulos, A. Budovski, C. Diaconu, A. Hernandez Saenz, J. Hu, H. Kodavalla, D. Kossmann, S. Lingam, U.F. Minhas, N. Prakash, V. Purohit, H. Qu, C.S. Ravellam, K. Reisteter, S. Shrotri, D. Tang, V. Wakade, Socrates: The new sql server in the cloud, in: Proceedings Of The 2019 International Conference On Management Of Data, 2019, pp. 1743–1756.

[14] F. Li, Cloud-native database systems at Alibaba: Opportunities and challenges, Proc. VLDB Endow. 12 (12) (2019) 2263–2272.

[15] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall, W. Vogels, Dynamo: Amazon's highly available key-value store, Oper. Syst. Rev. 41 (6) (2007) 205–220.

[16] F. Chang, J. Dean, S. Ghemawat, W.C. Hsieh, D.A. Wallach, M. Burrows, T. Chandra, A. Fikes, R.E. Gruber, Bigtable: A distributed storage system for structured data, ACM Trans. Comput. Syst. (TOCS) 26 (2) (2008) 1–26.

[17] L. George, HBase: The Definitive Guide: Random Access to Your Planet-Size Data, " O'Reilly Media, Inc.", 2011.

[18] A. Pavlo, M. Aslett, What's really new with NewSQL? ACM Sigmod Rec. 45 (2) (2016) 45–55.

[19] D. Bitton, D.J. DeWitt, C. Turbyfill, Benchmarking database systems-A systematic approach, Tech. rep., University of Wisconsin-Madison Department of Computer Sciences, 1983.

[20] C. Turbyfill, C.U. Orji, D. Bitton, AS3AP: An ANSI SQL standard scaleable and portable benchmark for relational database systems, in: The Benchmark Handbook 1993, 1993.

[21] A. Dey, A. Fekete, R. Nambiar, U. Röhm, YCSB+ T: Benchmarking web-scale transactional databases, in: 2014 IEEE 30th International Conference On Data Engineering Workshops, IEEE, 2014, pp. 223–230.

[22] B.F. Cooper, A. Silberstein, E. Tam, R. Ramakrishnan, R. Sears, Benchmarking cloud serving systems with YCSB, in: Proceedings Of The 1st ACM Symposium On Cloud Computing, 2010, pp. 143–154.

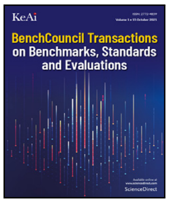
[23] TATP, http://tatpbenchmark.sourceforge.net.

[24] D. Bitton, M. Brown, R. Catell, S. Ceri, T. Chou, D. DeWitt, D. Gawlick, H. Garcia-Molina, B. Good, J. Gray, P. Homan, B. Jolls, T. Lukes, E. Lazowska, J. Nauman, M. Pong, A. Spector, K. Trieber, H. Sammer, O. Serlin, M. Stonebraker, A. Reuter, P. Weinberger, A measure of transaction processing power, Datamation 31 (7) (1985) 112–118.
[25] TPC-A, http://tpc.org/tpca/default5.asp.
[26] TPC-C, http://tpc.org/tpcc/default5.asp.
[27] M. Alomari, M. Cahill, A. Fekete, U. Rohm, The cost of serializability on platforms that use snapshot isolation, in: 2008 IEEE 24th International Conference On Data Engineering, IEEE, 2008, pp. 576–585.
[28] C. Zhang, Y. Li, R. Zhang, W. Qian, A. Zhou, Benchmarking on intensive transaction processing, Front. Comput. Sci. 14 (5) (2020) 1–18.
[29] V. Reniers, D. Van Landuyt, A. Rafique, W. Joosen, On the state of nosql benchmarks, in: Proceedings Of The 8th ACM/SPEC On International Conference On Performance Engineering Companion, 2017, pp. 107–112.
[30] S. Friedrich, W. Wingerath, F. Gessert, N. Ritter, E. Pldereder, L. Grunske, E. Schneider, D. Ull, NoSQL OLTP benchmarking: A survey, in: GI-Jahrestagung, 2014, pp. 693–704.
[31] R. Han, L.K. John, J. Zhan, Benchmarking big data systems: A review, IEEE Trans. Serv. Comput. 11 (3) (2017) 580–597.
[32] M. Barata, J. Bernardino, P. Furtado, Survey on big data and decision support benchmarks, in: International Conference On Database And Expert Systems Applications, Springer, 2014, pp. 174–182.
[33] X. Qin, X. Zhou, A survey on benchmarks for big data and some more considerations, in: International Conference On Intelligent Data Engineering And Automated Learning, Springer, 2013, pp. 619–627.
[34] A. Bonifati, G. Fletcher, J. Hidders, A. Iosup, A survey of benchmarks for graph-processing systems, in: Graph Data Management, Springer, 2018, pp. 163–186.
[35] D.F. Bacon, N. Bales, N. Bruno, B.F. Cooper, A. Dickinson, A. Fikes, C. Fraser, A. Gubarev, M. Joshi, E. Kogan, A. Lloyd, S. Melnik, R. Rao, D. Shue, C. Taylor, M. van der Holst, D. Woodford, Spanner: Becoming a SQL system, in: Proc. SIGMOD 2017, 2017, pp. 331–343.
[36] H.-T. Kung, J.T. Robinson, On optimistic methods for concurrency control, ACM Trans. Database Syst. (TODS) 6 (2) (1981) 213–226.
[37] P.A. Bernstein, V. Hadzilacos, N. Goodman, Concurrency Control And Recovery In Database Systems, Vol. 370, Addison-wesley Reading, 1987.
[38] S.S. Kulkarni, M. Demirbas, D. Madappa, B. Avva, M. Leone, Logical physical clocks, in: International Conference On Principles Of Distributed Systems, Springer, 2014, pp. 17–32.
[39] M. Raynal, M. Singhal, Logical time: Capturing causality in distributed systems, Computer 29 (2) (1996) 49–56.
[40] H. Lan, Z. Bao, Y. Peng, A survey on advancing the dbms query optimizer: Cardinality estimation, cost model, and plan enumeration, Data Sci. Eng. 6 (1) (2021) 86–101.
[41] S. Gilbert, N. Lynch, Perspectives on the CAP theorem, Computer 45 (2) (2012) 30–36.
[42] K.P. Birman, D.A. Freedman, Q. Huang, P. Dowell, Overcoming cap with consistent soft-state replication, Computer 45 (2) (2012) 50–58.
[43] A. Quamar, K.A. Kumar, A. Deshpande, SWORD: scalable workload-aware data placement for transactional workloads, in: Proceedings Of The 16th International Conference On Extending Database Technology, 2013, pp. 430–441.
[44] E. Zamanian, C. Binnig, A. Salama, Locality-aware partitioning in parallel database systems, in: Proceedings Of The 2015 ACM SIGMOD International Conference On Management Of Data, 2015, pp. 17–30.
[45] Y. Cheng, P. Ding, T. Wang, W. Lu, X. Du, Which category is better: Benchmarking relational and graph database management systems, Data Sci. Eng. 4 (4) (2019) 309–322.
[46] P. Gupta, M.J. Carey, S. Mehrotra, o. Yus, Smartbench: A benchmark for data management in smart spaces, Proc. VLDB Endow. 13 (12) (2020) 1807–1820.
[47] J. Kuhlenkamp, M. Klems, O. Röss, Benchmarking scalability and elasticity of distributed database systems, Proc. VLDB Endow. 7 (12) (2014) 1219–1230.
[48] J. Moeller, Z. Ye, K. Lin, W. Lang, Toto–benchmarking the efficiency of a cloud service, in: Proceedings Of The 2021 International Conference On Management Of Data, 2021, pp. 2543–2556.
[49] Micro Benchmark, https://hpc-wiki.info/hpc/Micro_benchmarking.
[50] Macro Benchmark, https://www.informit.com/articles/article.aspx?p=2144597&seqNum=2.
[51] A. Thomson, T. Diamond, S.-C. Weng, K. Ren, P. Shao, D.J. Abadi, Calvin: fast distributed transactions for partitioned database systems, in: Proceedings Of The 2012 ACM SIGMOD International Conference On Management Of Data, 2012, pp. 1–12.
[52] J.M. Faleiro, A. Thomson, D.J. Abadi, Lazy evaluation of transactions in database systems, in: Proceedings Of The 2014 ACM SIGMOD International Conference On Management Of Data, 2014, pp. 15–26.
[53] C. Xie, C. Su, M. Kapritsos, Y. Wang, N. Yaghmazadeh, L. Alvisi, P. Mahajan, Salt: Combining $\{ACID\}$ and $\{BASE\}$ in a distributed database, in: 11th $\{USENIX\}$ Symposium On Operating Systems Design And Implementation ($\{OSDI\}$ 14), 2014, pp. 495–509.
[54] TPC-E, http://tpc.org/tpce/default5.asp.
[55] D.J. DeWitt, The wisconsin benchmark: Past, present, and future, in: The Benchmark Handbook, J. Gray, Ed, 1993.
[56] D.J. DeWitt, C. Levine, Not just correct, but correct and fast: a look at one of Jim Gray's contributions to database system performance, ACM SIGMOD Rec. 37 (2) (2008) 45–49.
[57] M. Ferdman, A. Adileh, O. Kocberber, S. Volos, M. Alisafaee, D. Jevdjic, C. Kaynak, A.D. Popescu, A. Ailamaki, B. Falsafi, Clearing the clouds: a study of emerging scale-out workloads on modern hardware, Acm Sigplan Notices 47 (4) (2012) 37–48.
[58] P. Tözün, I. Pandis, C. Kaynak, D. Jevdjic, A. Ailamaki, From A to E: analyzing TPC's OLTP benchmarks: the obsolete, the ubiquitous, the unexplored, in: Proceedings Of The 16th International Conference On Extending Database Technology, 2013, pp. 17–28.
[59] D.E. Difallah, A. Pavlo, C. Curino, P. Cudre-Mauroux, Oltp-bench: An extensible testbed for benchmarking relational databases, Proc. VLDB Endow. 7 (4) (2013) 277–288.
[60] BenchmarkSQL, https://sourceforge.net/projects/benchmarksql.
[61] tpcc-mysql, https://github.com/Percona-Lab/tpcc-mysql.
[62] tpce-mysql, https://github.com/Percona-Lab/tpce-mysql.
[63] R.O. Nascimento, P.R. Maciel, Dbt-5: An open-source tpc-e implementation for global performance measurement of computer systems, Comput. Inf. 29 (5) (2010) 719–740.
[64] EGen, http://tpc.org/tpc_documents_current_versions/current_specifications5.asp.
[65] A. Pavlo, C. Curino, S. Zdonik, Skew-aware automatic database partitioning in shared-nothing, parallel OLTP systems, in: Proceedings Of The 2012 ACM SIGMOD International Conference On Management Of Data, 2012, pp. 61–72.
[66] C. Curino, E.P.C. Jones, Y. Zhang, S.R. Madden, Schism: a workload-driven approach to database replication and partitioning, Proc. VLDB Endow. (2010) 48–57.

Contents lists available at ScienceDirect

BenchCouncil Transactions on Benchmarks, Standards and Evaluations

journal homepage: https://www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/



Combating disinformation on social media: A computational perspective

Kai Shu

Department of Computer Science, Illinois Institute of Technology, Chicago 60616, United States of America

ARTICLE INFO

Keywords:
Disinformation
Social media mining
Social computing

ABSTRACT

The use of social media has accelerated information sharing and instantaneous communications. The low barrier to enter social media enables more users to participate and makes them stay engaged longer, while incentivizing individuals with a hidden agenda to use disinformation to manipulate information and influence opinions. Disinformation, such as fake news, hoaxes, and conspiracy theories, has increasingly been weaponized to divide people and create detrimental societal effects. Therefore, it is imperative to understand disinformation and systematically investigate how we can improve resistance against it, taking into account the tension between the need for information and the need for security and protection against disinformation. In this survey, we look into the concepts, methods, and recent advancements of detecting disinformation from a computational perspective. We will also discuss open issues and future research directions for combating disinformation on social media.

Contents

1. Introduction

Social media has become the leading platform for individuals to communicate [1]. In particular, social media is immensely popular for news dissemination, information sharing, and event participation due to its capacity to rapidly spread information at scale [2,3]. While this massive capacity can promote social trust and enhance social connectivity, it can also facilitate the rampant propagation of disinformation [4–6]. Such rampant disinformation often leverages the trust and social connectivity of social media users, spreads manipulated information to foment hatred, and inflicts damages on individuals or groups. With disinformation growing at unprecedented volumes on social media disinformation is now viewed as one of the greatest threats to democracy, justice, public trust, freedom of expression, journalism, and economic growth [4]. Hence, there is a pressing and necessary need to tackle digital disinformation.

However, detecting disinformation and fake news with computational approaches poses unique challenges that make it nontrivial. First, the *data challenge* has been a major roadblock because the content of fake news and disinformation is rather diverse in terms of topics, styles and media platforms; and fake news attempts to distort truth with diverse linguistic styles while simultaneously mocking true news. Thus, obtaining annotated fake news data is non-scalable, and data-specific embedding methods are not sufficient for fake news detection with little labeled data. Second, the *evolving challenge* of fake news and

E-mail address: kshu@iit.edu.

https://doi.org/10.1016/j.tbench.2022.100035
Received 3 December 2021; Received in revised form 22 March 2022; Accepted 23 March 2022
Available online 7 April 2022

Table 1
The comparison with representative fake news detection methods.

Representative Methods	News Content		Social Context			
	Linguistic	Visual	User	Post	Temporal	Network
[10–16]	✓					
[17,18]		✓				
[19–21]	✓	✓				
[22,23]			✓			
[24–26]				✓		
[27]				✓		✓
[28,29]					✓	✓
[30–32]						✓

disinformation is another obstacle in this task—fake news is usually related to newly emerging, time-critical events, which may not have been properly verified by existing knowledge bases due to the lack of corroborating evidence or claims. Third, the *explainability challenge* is concerned with the development of machine learning algorithms for disinformation that are explainable. Existing disinformation detection techniques are often machine learning black boxes that provide little or no explanation on the detection process. Explainability ensures that the developed algorithms are transparent, ensuring ethically responsible and trustworthy algorithms. However, deriving algorithmic explanations useful for domain experts and enhancing explanations by understanding and incorporating prior expert knowledge has been challenging.

We present the recent advancement of learning with weak social supervision to understand and detect disinformation and fake news on social media [7]. In particular, these methods are attempting diverse challenging scenarios towards detecting fake news more effectively, with explainability, at an early stage, and across domains. We further discuss the open issues and future directions along the line of combating disinformation from a computational perspective.

2. Related work and foundations

Disinformation and misinformation have been an important issue and attracts increasing attention in recent years [4,8,9]. The openness and anonymity of social media make it convenient for users to share and exchange information, but also makes it vulnerable to nefarious activities. Although the spread of misinformation and disinformation has been studied in journalism, the openness of social networking platforms, combined with the potential for automation, facilitates the dis/misinformation to rapidly propagate to massive numbers of people, which brings about unprecedented challenges. Specifically, disinformation is fake or inaccurate information that is intentionally spread to mislead and/or deceive; misinformation is false content shared by a person who does not realize that it is false or misleading.

As a representative example of disinformation, we briefly introduce the related work about fake news detection on social media. Fake news detection methods generally focus on using *news content* and *social context* [8,33] (as shown in Table 1).

News content contains the clues to differentiate fake and real news. News content-based features are the most explicit clues for fake news detection, given that the social media news evaluated are primarily textual in nature. The prerequisite of news content-based fake news detection is that the content of fake news should be somewhat different from truth in some quantifiable way [10]. For news content based approaches, features are extracted as linguistic-based and visual-based. Linguistic-based features capture specific writing styles and sensational headlines that commonly occur in fake news content [11], such as lexical and syntactic features. However, these methods have difficulty not only in generalizing hand-crafted linguistic features across topics, languages, and domains but also in utilizing the rich semantic and contextual information. To address the drawbacks of linguistics-based methods, the deep neural networks-based methods such as recurrent neural network (RNN) [12,29], convolutional neural networks (CNN) [13,14], and variational autoencoders (VAEs) [15,16] have been widely explored in recent years due to their capabilities to automatically learn latent textual representation and capture complex contextual patterns of news content.

Visual-based features try to identify fake images [17] that are intentionally created or capturing specific characteristics for images in fake news. News with visual information is likely to attract much more attention from social media users and thus gains a greater range of information dissemination [18]. Combining visual and linguistic features have shown better performances than using a single modality of feature. For example, Jin et al. first proposed a RNN-based automatic multimodal fake news detection model to fuse the visual and textual information of the post using an attention mechanism [19]. In addition, Zhou et al. presented a novel fake news method considering the correlations across the modalities [20].

In addition to news content, *social context* related to news pieces contains rich information to help detect fake news. For social context based approaches, the features mainly include user-based, post-based and network-based. User-based features are extracted from user profiles to measure their characteristics and credibilities [22,34,35]. For example, Shu et al. [22] proposed to understand user profiles from various aspects to differentiate fake news. Yang et al. [23] proposed an unsupervised fake news detection algorithm by utilizing users' opinions on social media and estimating their credibilities.

Post-based features represent users' social response in term of stance [24], topics [25], or credibility [29,34]. Network-based features are extracted by constructing specific networks, such as diffusion network [27,36], bipartite user–news interaction networks [37], etc. Propagation-based models assume that the credibility of news is highly related to the credibilities of relevant social media posts, which several propagation methods can be applied [24]. Recently, geometric deep learning such as Graph Neural Networks (GNNs) have been exploited to detect fake news and shown promising performances [28,30,31]. For example, Nguyen et al. proposed an inductive heterogeneous graph representation framework, Factual News Graph (FANG), which can effectively exploit social structure and engagement patterns of users for fake news detection [32]. Deep learning models are also applied to learn the temporal and linguistic representation of news [21,38]. Research also focuses on challenging problems of fake news detection, such as fake news early detection by adversarial learning [21] and user response generating [26].

Despite the success of aforementioned fake news detection algorithms, they mostly rely on large amounts of labeled instances to train supervised models. Such large labeled training data is often difficult to obtain for disinformation and fake news. Therefore, novel algorithms to learn with weak social supervision are needed for detecting fake news effectively, with explanation, at an early stage, and across domains.

3. Advancing disinformation detection

In this section, we will detail the proposed computational approaches on disinformation detection. We will first introduce a benchmark data repository for fake news detection, then describe four computational tasks on detecting disinformation. These four tasks

Table 2
The comparison with representative fake news detection datasets [39].

Datasets	News Content		Social Context				Spatiotemporal	
	Linguistic	Visual	User	Post	Response	Network	Spatial	Temporal
BuzzFeedNews	✓							
LIAR	✓							
BS Detector	✓							
CREDBANK	✓		✓	✓			✓	✓
BuzzFace	✓			✓	✓			✓
FacebookHoax	✓		✓	✓	✓			
NELA-GT-2018	✓							
FakeNewsNet	✓	✓	✓	✓	✓	✓	✓	✓

address the important aforementioned challenges of data, detection and evolving, and establish a principled way of learning with weak social supervision for social computing. Specifically, we aim to answer the following questions. First, when we have enough labeled data of news content and social context, how to detect fake news more effectively? Second, to involve and benefit domain experts, such as fact-checkers and journalists, how can we make fake news detection results more understandable? Third, social context may be useful to help detect fake news while it takes a long time to aggregate; can we detect fake news at an early stage and how early can it be? At last, fake news has diverse topics and obtaining labels for each domain is costly; would auxiliary information across domains be helpful in detecting fake news? For each of the task, we will present the technical details of the research issues and proposed research.

3.1. *FakeNewsNet: A Benchmark data repository*

The first and most important step to detect fake news is to collect a benchmark dataset. Despite several existing computational solutions on the detection of fake news, the lack of comprehensive and community-driven fake news datasets has become one of major roadblocks. Therefore, we create, and curate a multi-dimensional data repository *FakeNewsNet*,[1] which contains two datasets with news content, social context, and spatiotemporal information [39]. It currently contains two datasets with (23K) news pieces annotated from fact-checking sites, rich user engagements (691K users, 2M tweets and 2B network followers) from Twitter.

From Table 2, we observe that no existing public dataset can provide all possible features of news content, social context, and spatiotemporal information. The constructed FakeNewsNet repository has the potential to boost the study of various open research problems related to fake news study. First, the rich set of features in the datasets provides an opportunity to experiment with different approaches for fake new detection, understand the diffusion of fake news in social network and intervene in it. Second, temporal information enables the study of early fake news detection. Third, we can investigate the fake news diffusion process by identifying provenances, persuaders, and developing better fake news intervention strategies [40].

3.2. *Effective fake news detection*

In our recent work [41], we investigate effective fake news detection with social context. The basic idea is that the news ecosystem on social media provides abundant social context information, which involves three basic entities, i.e., publishers, news pieces, and users. Fig. 1 gives an illustration of such ecosystem. In Fig. 1, p_1, p_2 and p_3 are news publishers who publish news $a_1, \dots, a_4$ and $u_1, \dots, u_6$ are users who have engaged in sharing these news pieces. In addition, users tend to form social links with like-minded people with similar interests. The *tri-relationship*, the relationship among publishers, news pieces, and users, contains additional information to help detect fake news.

[1] https://github.com/KaiDMML/FakeNewsNet.

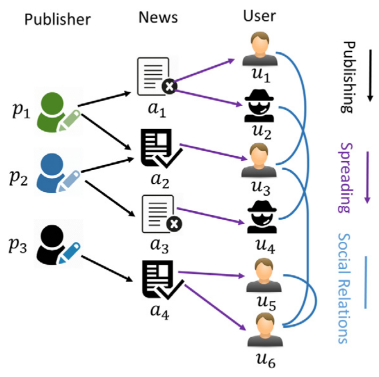


Fig. 1. An illustration of tri-relationship among publishers, news pieces, and users, during the news dissemination process [41].

We propose to use non-negative matrix factorization (NMF) to learn the news representations by encoding the relations in the heterogeneous network with the following objective function:

$$\min_{\mathbf{D},\mathbf{U},\mathbf{V},\mathbf{T}\geq 0,\mathbf{p},\mathbf{q}} \|\mathbf{X}-\mathbf{D}\mathbf{V}^T\|_F^2 + \alpha\|\mathbf{Y}\odot(\mathbf{A}-\mathbf{U}\mathbf{T}\mathbf{U}^T)\|_F^2 + \beta\mathrm{tr}(\mathbf{H}^T\mathbf{L}\mathbf{H}) + \gamma\|\mathbf{e}\odot(\bar{\mathbf{B}}\mathbf{D}\mathbf{q}-\mathbf{o})\|_2^2 + \eta\|\mathbf{D}_L\mathbf{p}-\mathbf{y}_L\|_2^2 + \lambda R \tag{1}$$

The proposed framework consists of three major components. First, the news content embedding learns news representation $\mathbf{D}$ by factorizing the bag-of-word matrix $\mathbf{X}$. Second, the social context embedding includes three parts, a user–user embedding from social relations $\mathbf{A}$, a user–news embedding from spreading relations $\mathbf{H}$, and a publisher–news embedding from publishing relation $\mathbf{B}$. Specifically, we decompose the matrix $\mathbf{A}$ into a user representation matrix $\mathbf{U}$, and $\mathbf{T}$ is the user–user correlation matrix, $\mathbf{Y}$ controls the contribution of $\mathbf{A}$, and $\odot$ denotes the Hadamard product operation. For the user–news embedding, we optimize the graph Laplacian of a matrix $\mathbf{H}$ which consists of user and news representation, and $\mathbf{L}$ is the Laplacian matrix. For the publisher–news embedding, the idea is to utilize publisher partisan labels vector $\mathbf{o}$ and the normalized publisher–news matrix $\bar{\mathbf{B}}$ to optimize the news feature matrix $\mathbf{D}$, where $\mathbf{e}$ is a binary matrix that indicates the label availability of publishing relations. Third, we further incorporate a semi-supervised linear classifier to map news representation $\mathbf{D}$ to news label $\mathbf{y}_L$ with the mapping function $\mathbf{p}$; R is a regularization terms to avoid overfitting, and α, β, γ, η, and λ are controlling the importance of each component. Experimental results show that TriFN has better fake news detection performance than state-of-the-art approaches. More details can be found in [41].

3.3. *Explainable fake news detection*

In [42,43], we study the explainable fake news detection. Despite the promising results of existing fake news detection methods, however, the majority of these methods focus on *detecting* fake news effectively with latent features but cannot explain "why" a piece of news was detected as fake news, which like a black-box. Being able to *explain*

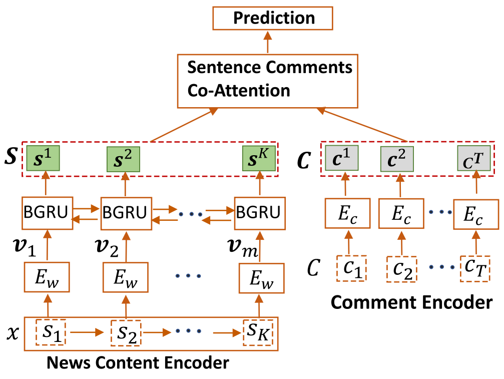


Fig. 2. The proposed framework dEFEND for explainable fake news detection [42].

why news was determined as fake from news contents and auxiliary information is much desirable because: (1) the derived explanation can provide new insights and knowledge originally hidden to practitioners; and (2) extracting explainable features from noisy auxiliary information can further help improve fake news detection performance. We aim to tackle the following challenges: (1) how to perform explainable fake news detection that can improve detection performance and explainability simultaneously; (2) how to extract explainable comments without the ground truth during training; and (3) how to model the correlation between news contents and user comments jointly for explainable fake news detection. The basic idea is that news content may contain information that is verifiably false and user comments express users' opinions, stances, and sentiment towards news content. News contents and user comments are inherently *related* to each other and can provide important clues to detect and explain why a given news article is fake. Thus, we simultaneously capture top-k check-worthy news sentences and user comments for fake news detection and explanation.

Fig. 2 shows the proposed dEFEND. In the figure, $x = \{s_1, \dots, s_K\}$ is the content of a news. $C = \{c_1, \dots, c_T\}$ is the set of comments on the news. dEFEND consists of news content encoder, comment encoder, sentence-comment co-attention layer and the prediction layer. The *news content encoder* designs an attentive hierarchical GRU to learn representations of each sentences, denoted as $\mathbf{S} = \{\mathbf{s}^1, \dots, \mathbf{s}^K\}$. Similarly, *comment encoder* uses LSTM to learn representations of comments, denoted as $\mathbf{C} = \{\mathbf{c}^1, \dots, \mathbf{c}^T\}$. With $\mathbf{S}$ and $\mathbf{C}$, the *sentence-comment co-attention layer* models the mutual influences between the news sentences and user comments to assign larger attention-weights to sentences and comments that can be used to detect and explain why a piece of news is fake. The final representation by aggregating representations of important news sentences and user comments is used for classification. Experimental results show that dEFEND has better fake news detection performance than state-of-the-art approaches. Meanwhile, dEFEND can derive meaningful interpretations from news content and comments. More details can be found in [42].

3.4. Early fake news detection

In our recent study, we exploit limited information from multiple sources on user comments (i.e., *weak signals*), along with text content, to detect fake news [44]. We observe that users' posts and comments carry rich crowd information including opinions, stances, and sentiments that can help detect fake news for various reasons. First, previous work has shown that conflicting sentiments among spreaders may indicate fake news [8,24]. Second, different users have different levels of credibility. These findings have great potential to provide additional signals for early detection of fake news. Thus, one can exploit multiple sources of weak social supervision simultaneously (in the form of weak labels) from social media to detect fake news.

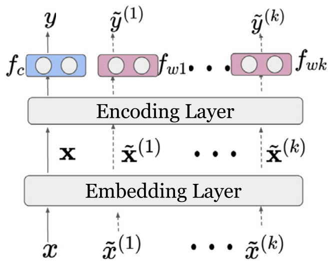


Fig. 3. Learning with weak signals from users' comments to detect disinformation [44].

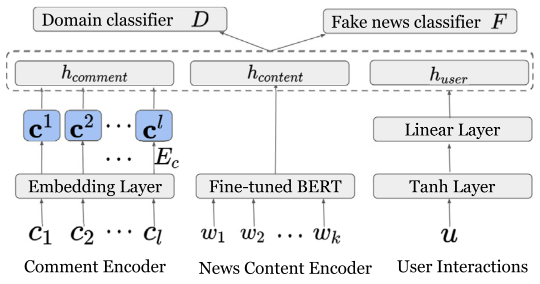


Fig. 4. Cross-domain Fake News Detection [45].

To that end, we have developed a framework MWSS jointly exploit Multiple sources of Weak Social Supervision (see Fig. 3). It utilizes a label weighting network (LWN) to model the weight of these weak labels $(\tilde{y}^{(1)}, \dots, \tilde{y}^{(k)})$ to regulate the learning of the fake news classifier f_x. The LWN is a meta-model to produce weights for the weak labels and can be trained by back-propagating the validation loss of a trained classifier on a separate dataset. LWN is suitable for early detection of fake news as LWN only requires news content for prediction. Our results show that weak supervision from multiple sources provides complementary information on top of news content; hence, can significantly improve fake news detection. More details can be found in [44].

3.5. Cross-domain fake news detection

In [45], we study cross-domain fake news detection. We address the challenge of limited within-domain labeled data for high-performance fake news detection by leveraging on cross-domain knowledge transfer and within-domain joint learning. We introduce a principled framework, CrossFND (Cross-domain Fake News Detection) (see Fig. 4). First, we utilize the embedding layer to encode each comment $c_1, \dots, c_l$ to obtain the comment representation $h_{comment} = [\mathbf{c}^1; \cdots; \mathbf{c}^l]$, fine-tuning BERT with the training data in the source and target domain such that it learns news representation $h_{content}$, and pass the user–news interaction vector u to a nonlinear and linear layer and get h_{user}. Then these representation vectors are concatenated and perform two tasks: (i) predicting whether a piece of news is fake or not with F; and (2) predicting which domain is the news coming from (i.e., source or target) with D. The final objective is a minmax function: $\min_F \max_D L(F) - \alpha L(D)$; where $L(D)$ $(L(F))$ is the cross-entropy loss to predict domain (news) label. Because we require the model to learn domain-independent features, we maximize domain loss $L(D)$. This forces the model to create a representation for news contents that represses domain-specific features; thus, it tries to deceive the classifier. To meet the objective of accurately classifying news as fake or authentic, we minimize the fake news classifier's loss function $L(F)$. More details can be found in [45].

4. Open issues and future research

Combating disinformation is an ongoing and continuous challenge in society. We discuss several emerging open issues for developing computational methods to detect and mitigate disinformation, highlighting the pressing need for interdisciplinary research.

4.1. Trustworthy AI for combating disinformation

People are reluctant to believe the results of AI-powered disinformation detection tools as these techniques are often like a black-box and lack of transparency. However, most of the current methodologies are data-driven and conducted in a passive fashion, and thus are inadequate to apprehend knowledge from human intents and demands. Thus, it is important to adapt learning strategies and acquire knowledge from human feedback. To establish the explainable methods for combating disinformation, it is important to investigate: (1) how to transform the meaningful human cognition into knowledge? (2) how to incorporate noisy, incomplete, and complicated human feedback for better representation learning? (3) how to interpret prediction results with knowledge reasoning and causal discovery. In addition, it is also critical to build equitable AI algorithms for detecting disinformation. For example, disinformation has been targeting marginalized groups, and it is important to understand how to measure the different types biases are and their degree in disinformation. Moreover, building a robust disinformation detector can increase its trustworthy and proactively considering adversarial attacks on disinformation detection methods can better guide the development of robust detection models.

4.2. Neural disinformation generation and detection

With the recent development of neural generation models such as GPT-3 [46] and BART [47], it becomes possible to generate realistic long documents such as news. There is a growing concern on these powerful language models being utilized by malicious users to generate disinformation. Detecting these neural fake news pieces firstly requires us to understand the characteristic of these news pieces and the detection difficulty. Some recent work propose to generate topic-preserving [48] and factual-enhanced [49] synthetic news pieces to understand the characteristics of machine-generated news. Though recent advancements on computationally detecting human-written disinformation show some promising results, it is largely unclear whether these models can distinguish machine-written disinformation effectively. It is important to explore: (1) how to generate realistic fake news with neural generative models to better understand neural-generated fake news? and (2) how is the capacity of human and machine on differentiating human-generated and machine-generated fake/real news?

4.3. Online disinformation and its offline impact

Disinformation is widely spread in online social networks and has been shown to result in offline events. Existing social network studies are mostly separately conducted to understand user online behaviors and offline activities in cyber and physical space. However, there is a growing need to integrate studies of these online and offline space and investigate their interdependencies. There is a gap to identify and assess how an online disinformation piece triggers actual behavior changes in the offline space. To establish the fundamental understanding of this gap, it is essential to explore: (1) how to discover the behavioral dynamics that are significant for inducing behavior change among different types of social networks; (2) how to develop high-fidelity models to understand the behavioral dynamics of transfer and spread among online and offline communities; and (3) how to build connections and causal analysis to understand the effects of online disinformation to the offline real-world events. Interdisciplinary research is much desired from disciplines such as social science, psychology, and computer science to deeply understand the aforementioned questions.

5. Conclusion

With the increasing popularity of social media, more and more people consume news from social media instead of traditional news media. However, social media has also enabled the wide dissemination of disinformation. In this article, we explored the fake news problem by reviewing existing literature, and discuss recent advancements for computationally detecting fake news with social media data and analysis. We also further discuss the promising future directions for combating disinformation and the pressing need for interdisciplinary research.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This material is in part supported by XCMG American Research Corporation, United States of America and Google Cloud Research Credits Award, United States of America.

References

[1] A. Perrin, Social media usage, Pew Res. Cent. 125 (2015) 52–68.
[2] A. Al-Rawi, Viral news on social media, Digit. J. 7 (1) (2019) 63–79.
[3] E. Shearer, A. Mitchell, News Use Across Social Media Platforms in 2020, Pew Research Center, 2020.
[4] K. Shu, H. Liu, Detecting fake news on social media, Synth. Lect. Data Min. Knowl. Discov. 11 (3) (2019) 1–129.
[5] H. Allcott, M. Gentzkow, Social Media and Fake News in the 2016 Election, Tech. Rep., National Bureau of Economic Research, 2017.
[6] A. Guess, J. Nagler, J. Tucker, Less than you think: Prevalence and predictors of fake news dissemination on facebook, Sci. Adv. 5 (1) (2019) eaau4586.
[7] K. Shu, Understanding Disinformation: Learning with Weak Social Supervision (Ph.D. thesis), Arizona State University, 2020.
[8] K. Shu, A. Sliva, S. Wang, J. Tang, H. Liu, Fake news detection on social media: A data mining perspective, ACM SIGKDD Explor. Newsl. 19 (1) (2017) 22–36.
[9] K. Shu, A. Bhattacharjee, F. Alatawi, T.H. Nazer, K. Ding, M. Karami, H. Liu, Combating disinformation in a social media age, Wiley Interdiscip. Rev. Data Min. Knowl. Discov. 10 (6) (2020) e1385.
[10] K. Sharma, F. Qian, H. Jiang, N. Ruchansky, M. Zhang, Y. Liu, Combating fake news: A survey on identification and mitigation techniques, ACM Trans. Intell. Syst. Technol. (TIST) 10 (3) (2019) 1–42.
[11] M. Potthast, J. Kiesel, K. Reinartz, J. Bevendorff, B. Stein, A stylometric inquiry into hyperpartisan and fake news, 2017, arXiv preprint arXiv:1702.05638.
[12] T. Chen, X. Li, H. Yin, J. Zhang, Call attention to rumors: Deep attention based recurrent neural networks for early rumor detection, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2018, pp. 40–52.
[13] W.Y. Wang, "Liar, liar pants on fire": A new benchmark dataset for fake news detection, 2017, arXiv preprint arXiv:1705.00648.
[14] M.H. Goldani, R. Safabakhsh, S. Momtazi, Convolutional neural network with margin loss for fake news detection, Inf. Process. Manage. 58 (1) (2021) 102418.
[15] M. Cheng, S. Nazarian, P. Bogdan, Vroc: Variational autoencoder-aided multi-task rumor classifier based on text, in: Proceedings of the Web Conference 2020, 2020, pp. 2892–2898.
[16] D. Khattar, J.S. Goud, M. Gupta, V. Varma, Mvae: Multimodal variational autoencoder for fake news detection, in: The World Wide Web Conference, 2019, pp. 2915–2921.
[17] A. Gupta, H. Lamba, P. Kumaraguru, A. Joshi, Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy, in: WWW'13.
[18] P. Qi, J. Cao, T. Yang, J. Guo, J. Li, Exploiting multi-domain visual information for fake news detection, in: 2019 IEEE International Conference on Data Mining, ICDM, IEEE, 2019, pp. 518–527.
[19] Z. Jin, J. Cao, H. Guo, Y. Zhang, J. Luo, Multimodal fusion with recurrent neural networks for rumor detection on microblogs, in: Proceedings of the 25th ACM International Conference on Multimedia, 2017, pp. 795–816.
[20] X. Zhou, J. Wu, R. Zafarani, [... Formula...]: Similarity-aware multi-modal fake news detection, Adv. Knowl. Discov. Data Min. 12085 (2020) 354.
[21] Y. Wang, F. Ma, Z. Jin, Y. Yuan, G. Xun, K. Jha, L. Su, J. Gao, EANN: Event adversarial neural networks for multi-modal fake news detection, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM, 2018, pp. 849–857.

[22] K. Shu, S. Wang, H. Liu, Understanding user profiles on social media for fake news detection, in: MIPR, 2018.
[23] S. Yang, K. Shu, S. Wang, R. Gu, F. Wu, H. Liu, Unsupervised fake news detection on social media: A generative approach, in: AAAI, 2019.
[24] Z. Jin, J. Cao, Y. Zhang, J. Luo, News verification by exploiting conflicting social viewpoints in microblogs, in: AAAI, 2016.
[25] J. Ma, W. Gao, Z. Wei, Y. Lu, K.-F. Wong, Detect rumors using time series of social context information on microblogging websites, in: CIKM, 2015.
[26] F. Qian, C. Gong, K. Sharma, Y. Liu, Neural user response generator: Fake news detection with collective user intelligence, in: IJCAI, 2018.
[27] Y. Dou, K. Shu, C. Xia, P.S. Yu, L. Sun, User preference-aware fake news detection, 2021, arXiv preprint arXiv:2104.12259.
[28] C. Song, K. Shu, B. Wu, Temporally evolving graph neural network for fake news detection, Inf. Process. Manage. 58 (6) (2021) 102712.
[29] L. Wu, H. Liu, Tracing fake-news footprints: Characterizing social media messages by how they propagate, in: WSDM, 2018.
[30] Y.-J. Lu, C.-T. Li, GCAN: Graph-aware co-attention networks for explainable fake news detection on social media, 2020, arXiv preprint arXiv:2004.11648.
[31] A. Silva, Y. Han, L. Luo, S. Karunasekera, C. Leckie, Propagation2Vec: Embedding partial propagation networks for explainable fake news early detection, Inf. Process. Manage. 58 (5) (2021) 102618.
[32] V.-H. Nguyen, K. Sugiyama, P. Nakov, M.-Y. Kan, Fang: Leveraging social context for fake news detection using graph representation, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, 2020, pp. 1165–1174.
[33] X. Zhou, R. Zafarani, K. Shu, H. Liu, Fake news: Fundamental theories, detection strategies and challenges, in: WSDM, 2019.
[34] C. Castillo, M. Mendoza, B. Poblete, Information credibility on twitter, in: WWW, 2011.
[35] S. Kwon, M. Cha, K. Jung, W. Chen, Y. Wang, Prominent features of rumor propagation in online social media, in: ICDM, 2013.
[36] K. Shu, D. Mahudeswaran, S. Wang, H. Liu, Hierarchical propagation networks for fake news detection: Investigation and exploitation, in: Proceedings of the International AAAI Conference on Web and Social Media, Vol. 14, 2020, pp. 626–637.
[37] X. Zhou, R. Zafarani, Network-based fake news detection: A pattern-driven approach, ACM SIGKDD Explor. Newsl. 21 (2) (2019) 48–60.
[38] H. Karimi, P. Roy, S. Saba-Sadiya, J. Tang, Multi-source multi-class fake news detection, in: COLING, 2018.
[39] K. Shu, D. Mahudeswaran, S. Wang, D. Lee, H. Liu, FakeNewsNet: A data repository with news content, social context and dynamic information for studying fake news on social media, 2018, arXiv preprint arXiv:1809.01286.
[40] K. Shu, H.R. Bernard, H. Liu, Studying fake news via network analysis: Detection and mitigation, 2018, CoRR, arXiv:1804.10233.
[41] K. Shu, S. Wang, H. Liu, Beyond news contents: The role of social context for fake news detection, in: WSDM, 2019.
[42] K. Shu, L. Cui, S. Wang, D. Lee, H. Liu, Defend: Explainable fake news detection, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 395–405.
[43] L. Cui, K. Shu, S. Wang, D. Lee, H. Liu, Defend: A system for explainable fake news detection, in: Proceedings of the 28th ACM International Conference on Information and Knowledge Management, 2019, pp. 2961–2964.
[44] K. Shu, G. Zheng, Y. Li, S. Mukherjee, A.H. Awadallah, S. Ruston, H. Liu, Leveraging multi-source weak social supervision for early detection of fake news, 2020, arXiv:2004.01732.
[45] K. Shu, M. Ahmadreza, H. Liu, Cross-domain fake news detection on social media: A context-aware adversarial approach, in: Frontiers in Fake Media Generation and Detection, Springer, 2022.
[46] T.B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., Language models are few-shot learners, 2020, arXiv preprint arXiv:2005.14165.
[47] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer, Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, 2019, arXiv preprint arXiv:1910.13461.
[48] A. Mosallanezhad, K. Shu, H. Liu, Topic-preserving synthetic news generation: An adversarial deep reinforcement learning approach, 2020, arXiv preprint arXiv:2010.16324.
[49] K. Shu, Y. Li, K. Ding, H. Liu, Fact-enhanced synthetic news generation, in: Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence, 2021.

Contents lists available at ScienceDirect

BenchCouncil Transactions on Benchmarks, Standards and Evaluations

journal homepage: https://www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/

Challenges and recent advances in the design of real-time wireless Cyber-Physical Systems

Romain Jacob

ETH Zurich, Switzerland

A R T I C L E I N F O

Keywords:
Synchronous transmissions
Wireless Cyber-Physical Systems
Replicability
Real-time systems

A B S T R A C T

Cyber-Physical Systems (CPS) refer to systems where some intelligence is embedded into devices that interact with their environment. Using wireless technology in such systems is desirable for better flexibility, improved maintainability, and cost reduction, among others. Moreover, CPS applications often specify deadlines; that is, maximal tolerable delays between the execution of distributed tasks. Systems that guarantee to meet such deadlines are called real-time systems. In the past few years, a technique known as synchronous transmissions (ST) has been shown to enable reliable and energy efficient communication, which is promising for the design of real-time wireless CPS.

We identify at least three issues that limit the adoption of ST in this domain: (i) ST is difficult to use due to stringent time synchronization requirements (in the order of µs). There is a lack of tools to facilitate the implementation of ST by CPS engineers, which are often not wireless communication experts. (ii) There are only few examples showcasing the use of ST for CPS applications and academic works based on ST tend to focus on communication rather than applications. Convincing proof-of-concept CPS applications are missing. (iii) The inherent variability of the wireless environment makes performance evaluation challenging. The lack of an agreed-upon methodology hinders experiment reproducibility and limits the confidence in the performance claims. This paper synthesizes recent advances what address these three problems, thereby enabling significant progress for future applications of low-power wireless technology in real-time CPS.

1. Introduction

Cyber-Physical Systems (CPS) are understood as systems where "*physical and software components are deeply intertwined, each operating on different spatial and temporal scales, exhibiting multiple and distinct behavioral modalities, and interacting with each other in a myriad of ways that change with context*" [1]. The domains of application of CPS are very diverse: *e.g.*, robotics, distributed monitoring, process control, power-grid management [2–4].

It is important to realize that the design of CPS encompasses three main aspects, mapping to as many research fields, with their own purpose and goals: The *embedded hardware design* aims to extend the amount of computational resources available (*e.g.*, processing power, memory, sensors and actuators) while limiting the cost, form factor, and energy consumption of a device. The *communication*, either wired or wireless, aims to transmit messages between distributed devices efficiently; that is, quickly and using little energy. Finally, the *distributed system design* realizes the implementation of the CPS functions, such as *e.g.*, remote monitoring and control of distributed processes.

The goal of the overall design is to reliably provide the specified CPS functions. Achieving this goal relies on hardware and communication; however reaching "perfect" communication, such as 100% packet reception rate, is not a goal in itself; it is merely a mean to an end. What truly matters is to fulfill the system functionality. Typically, CPS design aims to provide end-to-end performance guarantees, such as meeting hard deadlines between the execution of distributed tasks; *e.g.*, between the start of a sensing task to the end of the corresponding actuation tasks (Fig. 1). Meeting such deadlines is called *providing real-time guarantees*.

The potential benefits of wireless communication for CPS applications are well-known and include simpler deployment and maintenance, cheaper operational costs, lighter weight [5]. Furthermore, wireless is the only viable option in application domains including highly mobile nodes, such as an automated warehouse with transport robots [6] or teams of drones [7]. However, CPS applications have challenging performance requirements [8], which are hard to fulfill with a wireless design.

E-mail address: jacobr@ethz.ch.
URL: https://www.romainjacob.net/.

https://doi.org/10.1016/j.tbench.2022.100036
Received 13 January 2022; Received in revised form 17 March 2022; Accepted 17 March 2022
Available online 7 April 2022

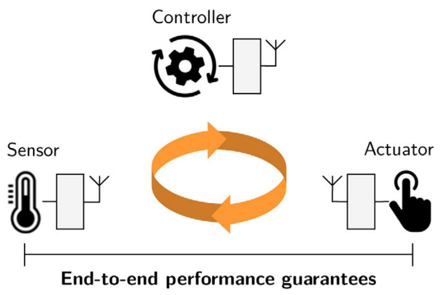


Fig. 1. The prime objective of CPS design is to provide end-to-end performance guarantees for distributed applications. In this paper, we consider synchronous transmissions, a recent development in low-power wireless communication, and demonstrate how to leverage the technique to provide real-time guarantees in wireless CPS.

1.1. Requirements of wireless CPS

CPS applications are subject to different types of requirements, such as the specified end-to-end latency, bandwidth, or number of devices; the precise performance level for these requirements depends on the application context. Generally, CPS requirements belong to one of the following classes:

Reliability A large ratio of messages is successfully transmitted wirelessly.

Adaptability The system adapts to runtime changes in resource demands.

Mobility The system supports mobile devices.

Timeliness Applications meet their deadlines, which are often specified end-to-end. Depending on the class of systems, deadlines can be either soft or hard [9].

Efficiency The system supports short end-to-end latency, scales in terms of system size, and optimizes its energy and bandwidth utilization.

These requirements are mutually conflicting. For example, reducing the energy consumption is typically achieved by keeping the radio turned off whenever possible. However, this directly conflicts with *Adaptability*, as the system cannot adapt reliably without exchanging some extra messages. In general, there is a price to pay in terms of *Efficiency* for meeting any of the other requirements. Hence, designing CPS consists in exploring the design space for relevant trade-offs; that is, the design optimizes the overall system *Efficiency* while meeting other application requirements.

1.2. Traditional wireless networking

Low-power wireless communication is a mature field of research, heavily studied for more than two decades. A large part of the research focused on wireless sensor networks, where low power consumption is a key requirement to enable long-term operation of the deployed networks, with specifications up to multiple years of operation on small batteries. Many successful applications and deployments include monitoring of soils [10], permafrost [11], buildings [12], or wildlife [13, 14].

In these scenarios, the distributed application often remains simple (*e.g.*, collect sensor readings). The main challenge is to reliably aggregate or disseminate messages across a multi-hop network. *Single-hop* communication refers to the case where a source node is in communication range from its destination. This is a rather simple case, but the deployed networks often span large areas whereas low-power radios can typically communicate in the range of tens of meters. Thus, *multi-hop* communication is required, whereby a source node must rely on other nodes in the network to forward its messages, hop after hop, until the destination is reached. This is the same principle as in the children's game known as Chinese whispers [15]; if you ever played, you know that the original message hardly ever reaches the end of the chain successfully.

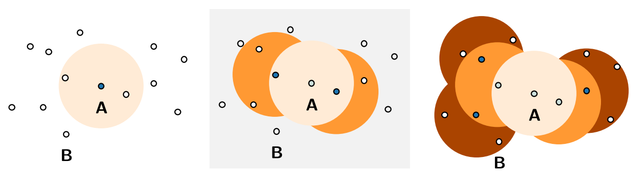
Multi-hop communication is a collaborative task for which the nodes must be coordinated. Indeed, if a node transmits a message while another wireless communication is ongoing, the transmissions will interfere and they may both fail. Furthermore, the radio frequency bands used for wireless communication cannot be isolated. Other networks are potentially exchanging messages on the same frequencies, which generates external interference and triggers packet losses. As a result, traditional multi-hop communication requires complex mechanisms for coordinating the nodes, scheduling the different transmissions to forward all messages throughout the network, and retransmitting messages that have been lost (*e.g.*, due to external interference). The complexity is further increased in mobile scenarios, where the set of neighboring nodes (which may relay a node's messages) changes frequently. The traditional wireless networking approach performs multi-hop communication by carefully planning a sequence of unicasts (*i.e.*, one-hop transmissions), usually performed along one or a few of the shortest paths possible between a message source and its destination [16–18]. Intuitively, this is efficient because only the necessary nodes are involved in relaying a message.

In practice however, multi-hop wireless network are sensitive to topology changes, external interference, and traffic congestion. These limit the reliability of communication, which has been a major obstacle to the utilization of wireless technology in CPS: for a long time, it has been considered impossible to provide the required level of reliability using wireless [19]. Synchronous transmissions have fundamentally changed that.

1.3. Synchronous transmissions

Synchronous transmissions (ST), also referred to as concurrent transmissions, is a technique consisting in letting multiple nodes transmit a message at the "same time" (hence the name of *synchronous* transmissions). A destination node can successfully receive (one of these) synchronous transmissions thanks to two effects taking place at the physical layer: constructive interference and the capture effect [20, 21]. In a nutshell, ST is likely to be successful if the incoming messages arrive at the receiving node's antenna within a small time offset (in the range of a few symbol periods—tens of µs—depending on the physical layer and the effect considered). ST has been shown to work both analytically [22], empirically [23], and on different physical layers, such as IEEE 802.15.4 [24], Bluetooth [25], and LoRa [26].

The use of ST in low-power communication, pioneered by Glossy [23] in 2011, has triggered a paradigm shift in the low-power wireless community: ST can be leveraged to implement efficient broadcast in a multi-hop network using network-wide flooding (Fig. 2). The flooding procedure implemented by Glossy is illustrated in Fig. 3. A first node initiates the flooding process. The 1-hop neighbors of the initiator receive the message and synchronously broadcast this same message in the next time step, which is then received by the initiator's 2-hop neighbors with high probability, thanks to ST. The process repeats following the same logic: a node that receives a packet broadcasts it again in the next time slot. Each node in the network transmits each packet up to N times, after which the flood terminates. It has been shown in a wide range of scenarios that, with $N = 3$, Glossy achieves a reliability above 99.99% [23]; that is, 99.99% of the floods are successfully received by nodes in the network. With $N = 5$, the average reliability reaches 99,999% [23]. Glossy achieves such high reliability by leveraging spatio-temporal redundancy. Packets are transmitted along all possible paths; in other words, they are implicitly routed everywhere, and therefore avoid interference sources localized in space. In addition, having each node transmitting N times creates temporal redundancy,



Node **A** initiates the flood. The 1-hop neighbors of **A** synchronously repeat. Node **B** receives the message.

Fig. 2. Flooding of a message from node **A** to node **B**.

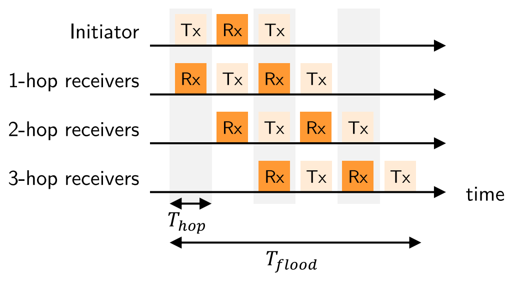


Fig. 3. Glossy operation in a 3-hop network with 2 transmissions per node (N).

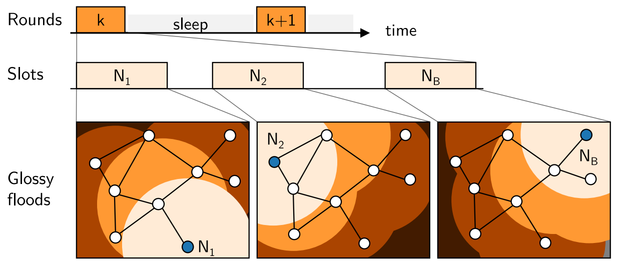


Fig. 4. Thanks to synchronous-transmissions-based flooding, a multi-hop network can be abstracted and scheduled like a shared bus. Communication is organized in rounds, composed of time slots; in each time slot, a node initiates a flood which allows to send a message to any other node(s) in bounded time. This mimics the operation of classical field bus, but with a wireless design.

thereby avoiding interference sources localized in time. Moreover, the predictability of the operation timing in ST-based flooding can be leveraged to perform distributed time synchronization. Glossy demonstrated that sub-µs synchronization accuracy can be achieved in a multi-hop network composed of tens to hundreds of nodes [23]. Since Glossy, other flooding strategies have been proposed [27–29], but the overall principle remains the same.

The key benefit of ST is that, thanks to the provided multi-hop broadcast primitive, the overall communication design can be drastically simplified. Essentially, one can abstract the underlying multi-hop topology as a *virtual single-hop network*, which can be scheduled like a shared bus: any node can send a message to any other node(s) in the network in bounded time. The only requirement is that no other node is using the "bus" at the same time. This design, first proposed with the Low-power Wireless Bus protocol [30], has been adapted into many flavors (see [31] for a recent survey) with always a similar concept: communication is organized in rounds, between which nodes keep their radio turned off to save energy. Each round is composed of time slots, which are assigned to certain nodes for communication. In each of these slots, nodes execute a flooding primitive (*e.g.*, Glossy) thereby preforming a one-to-all communication (Fig. 4). Consequently, the complexity of performing reliable multi-hop communication (see Section 1.2) is significantly relaxed. Thanks to ST, multi-hop communication is reduced to the scheduling of a single shared resource, a well-understood and relatively easy problem [9].

A priori, flooding seems to be a wasteful approach: every message sent by any node will be received and forwarded by every other node in the network. However, the simplicity and reliability of the approach actually pays off. (i) Since the flooding logic is simple, it requires little communication overhead for the coordination of the network; nodes mostly send application data. (ii) The spatio-temporal redundancy embedded in the flooding process makes it very reliable; once a flood is completed, there is hardly ever a need to further retransmit a message in a subsequent flood. (iii) Finally, since multiple nodes can transmit simultaneously, the flooding process completes quickly; very close to the theoretically optimal speed [23].

Thus, with flooding approaches based on ST, the energy cost of sending one byte of data is relatively high (since this byte will be retransmitted by all the nodes), but the *overall cost* for communication remains relatively small, thanks to the limited protocol overhead and the absence of need for further retransmissions. The energy efficiency and reliability of ST-based flooding has been demonstrated in many research contributions (*e.g.*, [23,32,33]) and showcased in the EWSN Dependability Competitions [34], where all wining solutions in the past four years (2016 to 2019) were based on ST [27,29,35–37].

The downside of ST is that it is difficult to use in more complex system designs, such as those envisioned for wireless CPS [8]. The difficulty stems from the tight timing requirements for successful ST: to be received reliably, transmissions must be initiated by the different nodes within few µs. Practically, this implies that the runtime execution of a node is governed by the communication protocol, which makes the implementation of advanced distributed tasks complex and error-prone. As a consequence, ST has thus far been mainly used for academic endeavors and mostly in wireless sensor network scenarios where the application tasks are typically simple and non-critical. Collecting a new sensor reading is a task that can usually tolerate being delayed by a few milliseconds while communication is ongoing. This is not acceptable for wireless CPS in general.

1.4. The dual-processor platform

In CPS, each device must perform application and communication tasks in order to fulfill the overall system functions; this poses the challenge of interference between tasks which contend for processor execution time. This interference problem can be mitigated by a new breed of embedded platforms featuring multiple processing cores, such as the NXP LPC541XX [38] or the VF3xxR [39]. On the one hand, this helps because applications and communication tasks can be processed in parallel, but on the other hand, it creates contention for the access to the resources shared between the cores. Efficient scheduling of multi-core platforms is a complex problem and a research field of its own.

Instead of resolving contention by scheduling, another approach proposed in the literature attempts to *prevent interference by design*. This principle, soberly called the Dual-Processor Platform (*DPP* [40]), consists in linking two processors with a processor interconnect called *Bolt* (Fig. 5). *Bolt* [41] provides predictable asynchronous message passing between two arbitrary processors while decoupling these processors with respect to time, power, and clock domains. The lower part of Fig. 5 shows a conceptual view of the *DPP*, including two message queues with first-in-first-out (FIFO) semantics, one for each direction, which are the only communication channels between the interconnected processors. The guiding principle of *Bolt* design is to limit the interference between the interconnected processors as much as possible, then to provide formally verified bounds on the unavoidable interference remaining. Concretely, this means that the *Bolt* API functions, used by the processors to exchange messages, have hard latency bounds. The upper part of Fig. 5 shows an early prototype of a *DPP*. Thanks to the separation of concerns between the scheduling and execution of application and communication tasks, the *DPP* concept provides an efficient and predicable architecture for CPS nodes. By

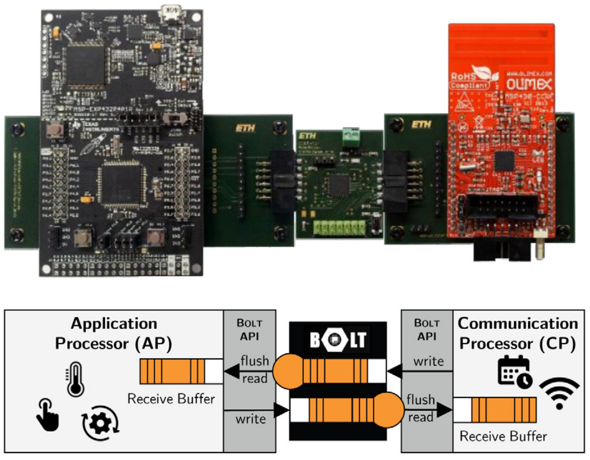


Fig. 5. Top.Example of a custom-built heterogeneous *DPP*. *Bolt (in the middle) interconnects a powerful application processor (TI MSP432* [42]) *on the left with a state-of-the-art communication processor (TI CC430* [43]) *on the right.* **Bottom.** Conceptual view of the *Bolt* processor interconnect. *Using the Bolt's API functions (*`write`*,* `read`*, and* `flush`*), the processors dedicated to application (AP)and communication (CP) can asynchronously exchange messages with predictable latency, while otherwise executing independently*.

entirely dedicating one processor to the application tasks and another one to wireless communication, we can decouple the timing of communication from the timing of the applications, and therefore facilitate the integration of ST in a CPS design. Furthermore, this helps to optimize performance: each processor can be customized for the specific operations it has to perform. The division of labor fosters specialization, thereby reducing the overall energy consumption and execution time; *i.e.*, maximizing the system's *Efficiency*.

1.5. *Performance evaluation in networking*

Over the past decade, low-power wireless communication has made significant progress, which are not limited to ST. The overall level of performance has increased, and it is now common to see reports of packet reception rates above 99% [23,44–46]. The more extreme the performance level, the more critical it becomes to confidently assess performance. Higher levels of confidence become necessary to argue about the differences in protocol design and quantify their performance trade-offs. Obviously, this is important for academics as it allow comparing competing approaches. But it is also important for industry: these new and promising technologies will never be adopted unless we can back up our performance claims confidently. In other words, others must be able to replicate our experiments.

In the context of wireless networking, replicable performance evaluation is made particularly challenging by the inherent variability of the experimental conditions: the uncontrollable dynamics of real-world networks [47,48] and the unsteady performance of hardware and software components [49,50] can cause a large variability in the experimental conditions, which makes it hard to quantitatively compare different solutions [51].

This reproducibility challenge (sometimes even referred to a "crisis" [52]) touches all scientific fields, and recently received significant attention in computer science [53–55]. Yet, how to practically design and execute performance evaluation experiments for wireless protocols remains a largely open question which is being debated by the community [56]. The lack of a standard for evaluating performance prevents a clear comparison of the different approaches, and therefore hinders the adoption of the technology. When everyone claims to be the best, one can hardly trust anyone.

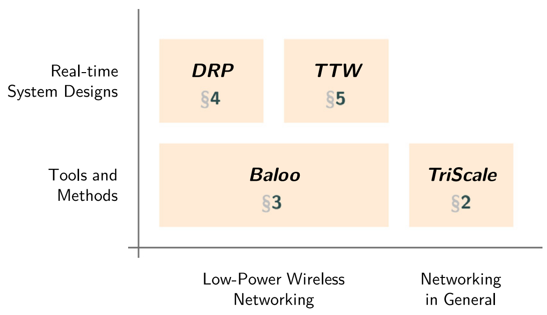


Fig. 6. Overview of the work presented in this paper.

1.6. *Recent advances*

In this paper, we summarize some of our recent work, in which we leverage recent advances in the domain of low-power wireless communication, in particular synchronous transmissions, in order to design wireless CPS providing end-to-end real-time guarantees. Fig. 6 provides a classification of the contributions, which we introduce below.

- We worked towards more rigorous and reproducible experimental networking research. In [57], we went beyond simple guidelines and proposed the first concrete methodology for designing networking experiments and analyzing their data. We leveraged this methodology to propose the first formalized definition of reproducibility for networking experiments. We implemented our methodology in *TriScale*, a first-of-its-kind tool that assists researchers by streamlining the design process and automating the data analysis (Section 2).
- We proposed and implemented *Baloo* [58], a design framework for network stacks based on synchronous transmissions (ST). *Baloo* significantly lowers the entry barrier for harnessing the efficiency, reliability and mobility support of ST: users implement their protocol through a simple yet flexible API while *Baloo* handles all the complex low-level operations based on the users' inputs (Section 3).
- We demonstrated for the first time that end-to-end real-time guarantees can be obtained in wireless CPS by leveraging the efficiency and reliability of synchronous transmissions. We proposed and implemented wireless real-time protocols for two different design objectives.
  - The Distributed Real-time Protocol (*DRP* [59]) uses contracts to maximize the flexibility of execution between application tasks (Section 4).
  - Time-Triggered Wireless (*TTW* [60]) statically co-schedules all task executions and message transfers to minimize end-to-end latency (Section 5).

The rest of this paper presents an overview of these contributions, highlights their underlying key ideas, then concludes with some directions for future work.

2. Designing replicable networking experiments with *TriScale*

The ability to replicate an experimental result is essential for making a scientifically sound claim. Without replicability[1] —the ability to

[1] Different terminology is used to refer to different aspects of replicability research [61,62]. In this paper, we refer to replicability as the ability of different researchers to follow the steps described in published work, collect new data using the same tools, and eventually obtain the same results, within the margins of experimental error. This is usually called replicability [63] but sometimes referred to as reproducibility.

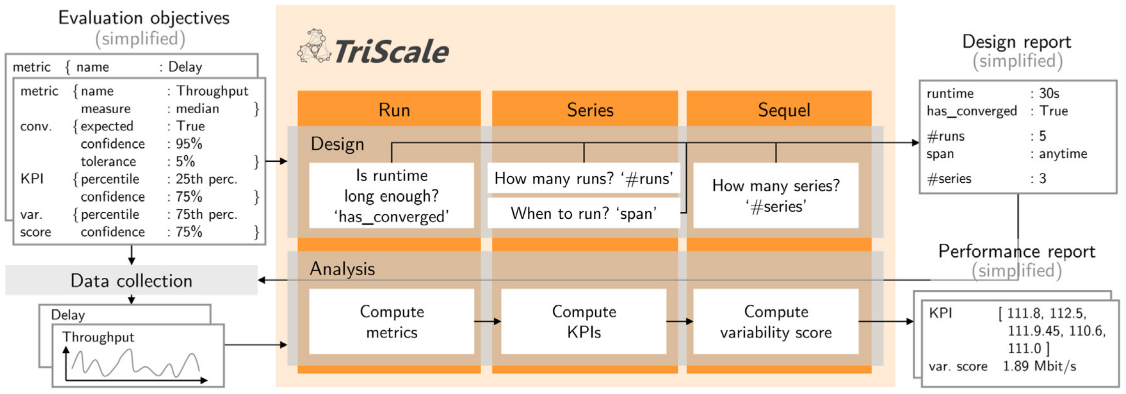


Fig. 7. Overview of *TriScale*. *TriScale is a framework supporting the design and analysis of networking experiments. TriScale assists the user in the design phase with a concrete methodology to answer important experiment design questions such as "How many runs?" and "How long should the runs be?" After the raw data are collected, TriScale supports the user by automating the data analysis. The framework implements robust statistics that handle the intrinsic variability of experimental networking data and returns expressive performance reports along with a variability score that quantifies the replicability of an experiment.*.

Table 1
A non-exhaustive list of factors hindering replicability, and selected networking references addressing them.

Focus of TriScale	
Variability in experiment design and data analysis	[54,55,64]
Other factors hindering replicability	
Lack of documentation	[55,65,66]
Artifacts unavailability/unusability	[53,67]
Uncontrollability of the exp. conditions	[47,48,68–70]
Variability of hardware and software behavior	[49,50,71]

assess the validity of claims reported by other researchers—any performance evaluation is questionable, at best. In networking, replicability is a well-recognized problem which stems from several factors (Table 1).

To be replicable, performance evaluations must account for the inherent variability of the experimental conditions—*i.e.*, the environment in which the experiment takes place—and the variability in hardware and software behavior in the system under test as well as in the measuring system. To facilitate this, the networking community has put great efforts into developing testbeds and data collection frameworks, *e.g.*, [68–70]. In addition, several calls for actions have been made to foster proper documentation [54,55] and artifact sharing [53,63] which are essential for replicability.

A more subtle but nonetheless important hindering factor for replicability are *differences in the methodology* used to design an experiment, analyze the resulting data, and draw conclusions from the evaluation. The literature related to this problem is currently limited to generic guidelines [54,55,72] and recommendations [64,66,73], which leave open several critical questions *before* (How many runs? How long should a run be?) and *after* experiments are conducted (How to process the data and analyze the results?). Without a concrete methodology, networking researchers often design and analyze similar experiments in different ways, making them hardly comparable [56]. Yet, strong claims are being made ("our system improves latency by 3×") while confidence is often discussed only in qualitative ways ("with high confidence"), if at all [70,71]. Furthermore, it is unclear how to effectively assess whether an experiment is indeed replicable. We argue that a concrete methodology is needed to help resolve this situation.

Hence, we developed such a methodology for the design and analysis of performance evaluations for networking research. In [57], we introduced *TriScale*, an implementation of our methodology into a software framework making the methodology readily applicable by researchers. While we do not claim that our methodology is fitting all situations, nor that it is the best one possible, we do find it useful in many practical cases. At a high level, our methodology features four key desirable properties.

Rationality The methodology rationalizes the experiment design by linking the design questions (*e.g.*, How many runs?) with the confidence in the performance claims.

Robustness The methodology is robust against the variability of the experimental conditions. The data analysis uses statistics that are compatible with the nature of networking data and are able to quantify the expected performance variation shall the evaluation be replicated.

Generality The methodology is applicable to a wide range of performance metrics, evaluation scenarios (emulator, testbed, in the wild), and network types (wired, wireless).

Conciseness The methodology describes the experimental design and the data analysis in a concise and unambiguous way to foster replicability while minimizing the use of highly treasured space in scientific papers.

Key idea. *TriScale*'s methodology is based on an analysis of the temporal characteristics of variability in networking experiments, which we argue can be decomposed into three timescales, which can be mapped to specific questions of the experiment design (see Fig. 7). For each timescale, *TriScale* applies a set of appropriate and rigorous statistical methods to derive performance results with *quantifiable confidence*. For each performance metric, *TriScale* computes a variability score that estimates, with a given confidence, how similar the results would be if the evaluation were replicated.
Limitations. With *TriScale*, we provide a concrete methodology that *concretely guides* networking researchers through the design of their experiments and the analysis of the gathered data, while *quantifying the replicability* of the performance evaluation. Hence, *TriScale* complements prior work toward replicable networking research that mostly focused on data collection, *e.g.*, [68–70].
Take-away. *TriScale* is implemented as a Python package [74]. For each timescale, a dedicated function takes raw data as input, performs the corresponding test or analysis, returns the results, and produces data visualizations. We aimed to make *TriScale* intuitive and easy to use. For a better impression of its usability, you can run an interactive demo directly in your web browser [75]. We expect *TriScale*'s open availability to actively encourage its use by the networking community and promote better experimentation practices in the short term.

The quest towards highly-reproducible networking experiments remains open, but we believe that *TriScale* represents an important stepping stone towards an accepted standard for experimental evaluations in networking.

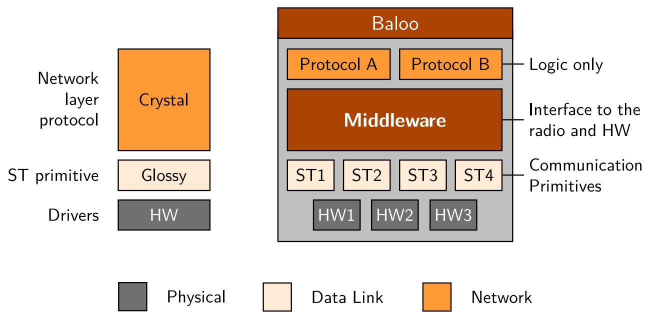
3. Synchronous transmissions made easy: Design your network stack with *Baloo*

As introduced in Section 1.3, Synchronous Transmissions (ST) is an increasingly used wireless communication technology for low-power multi-hop networks. Popularized by Glossy [23] in 2011, it has been proven to be highly reliable and energy efficient, as illustrated by the EWSN Dependability Competition [34], where all wining solutions were based on ST [27,29,35–37] (from 2016 to 2019).

A *ST primitive* refers to a protocol that efficiently realizes broadcast (*i.e.*, any-to-all communication) in bounded time, usually relying on *flooding*. Flooding is a communication strategy that realizes broadcast by having all receivers of a packet retransmit this same packet to all their neighbors; the packet is thus "flooded" through the whole network. ST makes flooding energy and time efficient by letting multiple wireless nodes transmit the packet *synchronously*, hence the name of *Synchronous Transmissions*. The successful reception of the packet can be achieved if the transmitters are tightly synchronized, thanks to *constructive interference* and the *capture effect* [20]. The synchronization requirements vary from sub- μs to tens of μs, depending on the platform and modulation scheme [20]. Such a broadcast primitive simplifies the design of network layer protocols: The underlying multi-hop network can be abstracted as a *virtual single-hop network* and thus be scheduled like a shared bus [30].

Since Glossy [23], many flavors of ST primitives have been proposed to improve performance in terms of reliability, latency, and energy consumption. To be more resilient to strong interference, Robust Flooding [27] is a primitive that modifies the RX-TX sequence from the original Glossy, whereas RedFixHop [76] uses hardware acknowledgments to minimize the number of retransmissions required. Instead, some primitives aim to minimize latency for specific traffic patterns. For example, Chaos [32] lets all nodes modify the packet being flooded to quickly aggregate information (*e.g.*, the max value of all sensor readings) or efficiently perform all-to-all data sharing to achieve distributed consensus [77]. Codecast [78] also targets many-to-many exchange for a larger amount of data. Pando [79] is another primitive focused on high throughput, which uses fountain code and packet pipelining for efficient data dissemination. Syncast [80] aims to reduce the radio on time required to save energy, while Less is More (LiM) [81] is a primitive that reduces energy consumption using learning to avoid unnecessary retransmissions during flooding.

All these primitives share the same drawback: Successful ST requires low-level control of timers and radio events in order to meet ST tight synchronization requirements (the order of μs). This degree of accuracy is difficult to achieve as it requires a detailed knowledge of the underlying hardware, low-level control of the radio operations, and a very careful management of software delays. As a result, designing a network stack based on ST is a complex and time-consuming task, for which only few solutions have been proposed. One of the first was the Low-power Wireless Bus (LWB) [30], which tries to flexibly support all kinds of traffic patterns in a balanced trade-off between latency and energy consumption. The same group designed eLWB [82], a variation of LWB tailored to event-based data collection. Sleeping Beauty [83] was later proposed to minimize energy consumption for data collection scenarios with many redundant sensor nodes. Time-Triggered-Wireless (*TTW* [60], Section 5) was designed to minimize the end-to-end latency between communicating application tasks. Finally, Crystal [46] has been proposed as a network stack specialized for sporadic data collection. All these network stacks solely rely on Glossy as ST primitive. In principle however, the same protocol logic could benefit from *multiple* primitives. For example, an LWB network could use Robust Flooding [27] in case of high interference, then revert to Glossy [23] for better time synchronization. If nodes need reprogramming, the software update can be quickly disseminated using Pando [79]. Designing a modular network stack supporting multiple ST primitives adds a new level of complexity.



(a) The implementation of the network layer protocol (Crystal) couples the interface to the underlying ST primitive (Glossy) and the protocol logic, *i.e.*, how long are the communication rounds, which radio channel is used, etc.

(b) Thanks to its additional middleware layer, *Baloo* flexibly supports multiple ST primitives and significantly reduces the efforts required to implement network layer protocols compared to traditional stacks, like LWB [34] or Crystal [46].

Fig. 8. Crystal [46] is a typical example of network stack based on ST (Fig. 8a). Conversely, *Baloo* is a flexible design framework. It is based on a middleware layer that separates the concern of timely execution of ST primitives from the implementation of the protocol logic (Fig. 8b).

Question 1 Can we facilitate the design of wireless network stacks based on Synchronous Transmission?

Question 2 Can we implement flexible and adaptive protocols, potentially leveraging multiple ST primitives, while guaranteeing that the timing requirements of ST are met?

The problem. To facilitate the network stack design (**Question 1**), a natural idea is to separate the concern of the timely execution of the primitives from the implementation of the protocol logic. One way to achieve such separation of concerns is to use a *middleware* as part of the network stack. The idea of a middleware for Wireless Sensor Networks (WSN) is not new, and the main challenge in such an endeavor is well-known. As phrased by Mottola and Picco [84], "*striking a balance between flexibility and complexity in providing access to low-level features is probably one of the toughest, yet most important, problems in WSN middleware*". The design of a middleware for ST is particularly challenging. Indeed, meeting the tight timing requirements for ST is directly conflicting with the concept of abstraction of a middleware: How to guarantee that the network layer does not hinder the timing accuracy for ST if it is itself unaware of the execution of the primitives? That is **Question 2**.

The challenge. A middleware for ST should meet the following requirements.

Usability The middleware must realize a well-defined interface enabling runtime control from the network layer (which implements the protocol logic) over the execution of the underlying ST primitives.

Generality The middleware must enable the implementation of a large variety of network layer protocols.

Versatility The middleware must enable one network layer protocol to use multiple ST primitives and switch between them at runtime.

Synchronicity The middleware must guarantee to respect the time synchronization requirements for ST (from sub- μs to tens of μs [20]).

Our solution. To address these challenges, we have designed *Baloo* [58],[2] a flexible design framework for low-power network stacks based on ST. *Baloo* provides a large set of features enabling performant protocol designs, while abstracting away low-level hardware management such as interrupt handling and radio core control. In summary:

- We proposed *Baloo*, a flexible design framework for low-power wireless network stacks based on ST (see Fig. 8).
- We presented the design of a middleware layer that meets all our requirements. This middleware forms the core component of *Baloo*.
- We showcased the usability of *Baloo* by re-implementing three well-known network stacks using ST: the Low-power Wireless Bus (LWB) [30], Sleeping Beauty [83], and Crystal [46].
- We illustrated the portability of *Baloo* by providing implementations for two platforms — the CC430 SoC [43] and the old but still heavily used TelosB mote [85].
- We demonstrated that *Baloo* induces only limited performance overhead (memory usage, radio duty cycle) compared to the original implementations.

Key idea. The core of *Baloo* is its clean API, based on callback functions, which let the users focus on implementing the protocol logic without worrying about low-level radio control (interrupt handling, timer settings, etc..). The API is generic and supports the different communication primitives. Through this API, multiple primitives can be used within the same network stack without additional complexity for the users.

Limitations. *Baloo* is a tool that facilitate the *implementation* of ST-based protocols, but it does not help to *design* them. The protocol design space is very large, with many trade-offs to consider depending on the application use case. This is still a fertile area of research, with recent proposals including [86–92].

Take-away. *Baloo* is openly available and is accompanied by a detailed documentation of its features and how to use them [93]. Our re-implementations of Crystal, Sleeping Beauty, and LWB are also available. We believe *Baloo* will be an important enabler for the development of real-world applications leveraging state-of-the-art ST technology.

4. *DRP*: Flexible real-time guarantees

As introduced in Section 1, Cyber-physical systems (CPS) tightly integrate components for sensing, actuating, and computing into distributed feedback loops to directly control physical processes [4]. As many CPS applications are mission-critical and physical processes evolve as a function of time, the communication among the sensing, actuating, and computing elements is often subject to real-time requirements, for example, to guarantee stability of the feedback loops [19]. These real-time requirements are often specified from an end-to-end application perspective. For example, a control engineer may require that sensor readings taken at time t are available for computing the control law at $t + D$, where the relative deadline D is derived from the application requirements; *e.g.*, the maximum tolerable delay between a sensing and a control task, where these tasks are typically executed on physically distributed devices.

Meeting end-to-end deadlines is non-trivial because data transfers between application tasks involves multiple other tasks (*e.g.*, operating system, networking protocols) and shared resources (*e.g.*, memories, system buses, wireless medium). The entire transmission chain of the data throughout the system must be taken into account to enable end-to-end real-time guarantees.

[2] The framework provides the "bare necessities" for the design and implementation of ST-based network stacks; so we called it *Baloo*.

Question 3 Can we provide end-to-end real-time guarantees between distributed applications in wireless CPS?

Question 4 Can we do so while preserving runtime adaptability and flexibility in the timing of task executions?

The problem. Enabling real-time communication between network interfaces of sources and destinations in a low-power wireless network has been studied for more than a decade [94–96]. Today, standards such as WirelessHART [97] and ISA100.11a [98] for control applications in the process industries already exist [99], and considerable progress in real-time transmission scheduling and end-to-end delay analysis for WirelessHART networks has been made [100,101].

Unfortunately, wireless real-time protocols such as WirelessHART [97] or Blink [102] only provide guarantees for message transmissions between *network interfaces*. These protocols do not handle the application schedules; at the source, the message release is typically assumed periodic; at the destination, nothing guarantees that the application will process the message in time. Providing end-to-end guarantees between *distributed application* (**Question 3**) demands to combine a wireless real-time protocol with the rest of the system; *i.e.*, consider application schedules and handle interference on shared resources.

The challenge. To support a broad spectrum of CPS applications, a solution to this problem should fulfill the following requirements.

Timeliness All messages received by the destination application meet their end-to-end deadlines.

Reliability All messages received at the wireless network interface are successfully delivered to their destination application (*i.e.*, no buffer overflows).

Adaptability The system adapts to dynamic changes in traffic requirements at runtime.

Composability Existing hardware and software components can be freely composed to satisfy the application's needs, without altering the properties of the integrated parts.

Efficiency The solution scales to large systems and operates efficiently with respect to resources such as energy, wireless bandwidth, computing capacity, and memory.

The main challenge consists in funneling messages in real-time through tasks that run concurrently and access shared resources. Interference on such resources can delay tasks and communication arbitrarily, therefore hampering *Timeliness*, *Reliability*, and *Composability*.

Our solution. In [59], we presented *DRP*, a real-time wireless CPS that tackles interference on shared resources by defining (minimal) constraints on the application schedules. This is achieved by combining a predictable device architecture with a real-time scheduler for the entire system.

Predictable device architecture We use the Dual-Processor Platform (*DPP*) concept (Section 1.4). The *DPP* dedicates a communication processor (CP) exclusively to the real-time network protocol and executes all other tasks on an application processor (AP). The *DPP* is based on the *Bolt* interconnect [41], which decouples two processors in the time, power, and clock domains, while allowing them to asynchronously exchange messages within predictable time bounds.

Thus, on each device, we decouple the communication and application tasks, which can be independently invoked in an event- or time-triggered fashion. The *DPP* concept guarantees the faithfulness of the network interface (*Reliability*), supports *Composability*, and leverages the recent trend toward ultra low-power multi-processor architectures, which can be chosen individually to match the needs of the application and the networking protocol respectively (*Efficiency*).

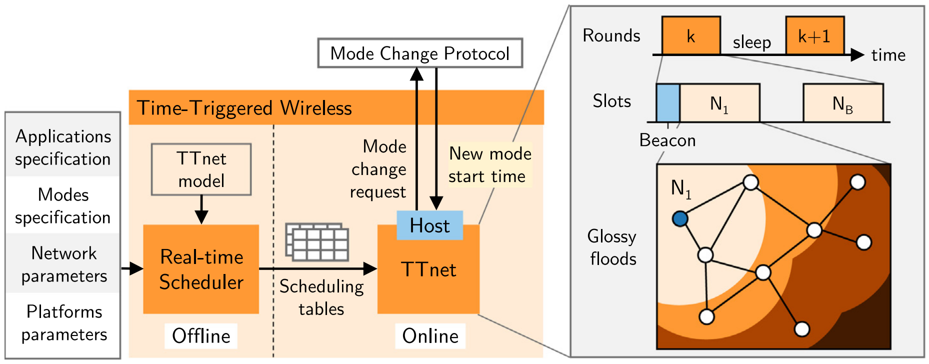


Fig. 9. High-level overview of the Time-Triggered Wireless (*TTW*) architecture.

Real-time scheduler We design the Distributed Real-time Protocol (*DRP*), a scheduler that provably guarantees that all messages received at the application interfaces meet their end-to-end deadlines (*Timeliness*) and that message buffers along the data transfers do not overflow (*Reliability*).

To accomplish this while being adaptive to unpredictable changes (*Adaptability*), *DRP* dynamically establishes at runtime a set of contracts based on the current traffic demands in the system. A contract determines the mutual obligations in terms of (i) minimum service provided, and (ii) maximum demand generated between the networking protocol and an application. *DRP* contracts define time bounds that can be analyzed to ensure that end-to-end deadlines are met, while preserving flexibility in the timing of distributed task executions (**Question 4**).

In summary, [59] presents the following contributions.

- We designed *DRP*, a wireless CPS system that provably provides end-to-end real-time guarantees between distributed applications. *DRP* does so by harnessing the benefits of synchronous transmissions (Section 1.3) and building upon the Blink real-time scheduler [102] and the Dual-Processor Platform architecture (Section 1.4).
- We simulated *DRP* execution to demonstrate that the provided bounds are both safe and tight, we implemented the protocol on embedded hardware, and showcased that it works as expected.
- We made our implementation of *DRP* publicly available [103], which includes the Blink scheduler for LWB [30].

Key idea. The key concept of *DRP* is to (i) physically decouple the communication protocol from the application tasks (each running on dedicated communication and application processors), and (ii) guarantee the timeliness of message transmissions throughout the system using minimally restrictive contracts between the different entities.
Take away. Our proof-of-concept implementation of *DRP* on embedded hardware confirmed that *DRP* appears to be a promising solution for low-rate applications, such as smart homes, where coexists multiple context-specific "applications" (*e.g.*, fridge, air-conditioning, lightning) which would particularly benefit from being scheduled independently of each other while being able to communicate in real-time.
Limitations. By design, enabling timing asynchrony between the connected applications leads to long end-to-end delays. Therefore, *DRP* is ill-suited for latency-sensitive applications, for which we designed a different system, presented in the next section.

5. *TTW*: Low-latency real-time guarantees

We revisited the challenge addressed in the previous section: Providing end-to-end real-time guarantees in wireless cyber–physical systems (CPS). With the design of *DRP* (Section 4), we demonstrated that, by leveraging synchronous transmissions (ST), it is possible to meet end-to-end deadlines between distributed tasks communicating through a multi-hop wireless network. *DRP* keeps all tasks as independent as possible; *i.e.*, constraining their schedule as little as necessary to provide end-to-end guarantees.

Because of that maximal-flexibility principle, the guarantees that can be provided by *DRP* are rather "slow": the minimal end-to-end deadline supported by the protocol is more than two times as large as a communication round. Furthermore, there is large jitter between successive task executions and message transmissions. This does not comply well with the requirements of industrial CPS applications, which often require short delays (the order of ms) and benefit from negligible jitter.

Thus, we considered another design objective: Instead of focusing on flexibility, we aim for minimizing latency and jitter in the system execution.

Question 3 Can we provide end-to-end real-time guarantees between distributed applications in wireless CPS?

Question 5 How can we minimize latency and jitter in the application execution while retaining some level of runtime adaptability?

The problem. To understand the challenges of wireless CPS, it is helpful to highlight the fundamental difference between a field bus and a wireless network. In a field bus, whenever a node is not transmitting, it can idly listen for incoming messages. Upon request from a central host, each node can wake up and react quickly. For a low-power wireless node, the major part of the energy is consumed by its radio. Therefore, energy efficiency requires to turn the radio off whenever possible to support long autonomous operation without an external power source. Since nodes are unreachable until they wake up, they require overlapping wake-up time intervals to communicate.

This observation often results in wireless system designs that minimize energy consumption by using communication rounds, *i.e.*, time intervals where all nodes wake-up, exchange messages, then turn off their radio [30,58,97,99]. Scheduling policies define when the rounds take place (*i.e.*, when to wake up) and which nodes are allowed to send messages during the round. Moreover, CPS do not only exchange messages, they also execute tasks (*e.g.*, sensing or actuation). Typically, the system requirements are specified end-to-end, *i.e.*, between distributed tasks exchanging messages. One option to meet such end-to-end requirements (**Question 3**) is to co-schedule the execution of tasks and the transmission of messages, as proposed in the literature for wired architectures [104–106]. However, these schedules result from complex optimization problems which are difficult to solve online, even more so in a low-power setting. Thus, schedules are often pre-computed offline, which restricts the runtime adaptability of the resulting system (**Question 5**).
The challenge. To support wireless CPS applications in an industrial context, a solution to this problem should fulfill the following requirements.

Timeliness All distributed applications meet their end-to-end deadlines.

Reliability A large ratio of messages is successfully transmitted over wireless and conflict-free communication is guaranteed between the system's nodes.

Adaptability The system adapts to runtime changes.

Mobility The system supports mobile devices.

Efficiency The system supports short end-to-end latency (ms), scales to medium-to-large system sizes, and optimizes its energy consumption and bandwidth utilization.

Our solution. In [60], we proposed *TTW*, a solution to the industrial wireless CPS problem that fulfill these requirements. We do so by combining co-scheduling techniques, inspired from the wired literature, with a ST-based wireless system design using communication rounds.

ST provides highly reliable wireless communication (*Reliability*) and inherent support for *Mobility*. A round-based design allows to minimize the energy consumed for communication, which is a large part of the total energy budget of a low-power system (*Efficiency*). The co-scheduling approach results in highly optimized schedules (*Efficiency*) which guarantee to meet the application deadlines (*Timeliness*). *TTW* provides some runtime *Adaptability* by switching between multiple pre-computed operation modes, a well-known concept in the wired literature [107].

The main challenge in realizing such a system is to integrate the allocation of messages to communication rounds (which is similar to a bin-packing problem [108]) with a co-scheduling approach (which typically solves a MILP [109] or an SMT [110–112] formulation). In summary, [60] presents the following contributions.

- We presented Time-Triggered Wireless (*TTW*, illustrated in Fig. 9), a low-power wireless CPS that meets the common requirements of industrial applications.
- We formulated a joint optimization problem for co-scheduling distributed tasks, messages, and communication rounds that guarantees to meet application deadlines, minimize the energy consumed for wireless communication, and ensures safety in terms of conflict-free communication, even under packet loss.
- We provided a methodology that efficiently solves this optimization problem, known to be NP-hard [113].
- Using time and energy models, we quantified the benefits of rounds to minimize energy, and we derive the minimum end-to-end latency achievable.
- We implemented *TTW* on embedded hardware and demonstrate that the system is suited for fast feedback control applications.

Key idea. The main challenge in the *TTW* design is that, with wireless communication, it is highly beneficial in terms of energy to send messages in rounds. Thus, the assignment of messages to round (similar to a bin-packing problem) must be combined to the traditional co-scheduling approaches, which is non-trivial. We solved this problem and implemented a multi-mode scheduler that allows critical applications to seamlessly switch between modes while minimizing the energy consumption spent for wireless communication. We further implemented a predictable network stack, called *TTnet*. Together, these two pieces form *TTW*, a publicly available [114] real-time wireless CPS design.
Limitations. Efficient static scheduling requires a precise model of the worst-case execution time of the tasks, which may be challenging to get in practice. Moreover, the synthesis of scheduling tables is computationally expensive, and does not scale well to large applications (hundreds of tasks and messages per application's period).
Take-away. *TTW* achieves near-optimal end-to-end latency by a tight coupling in the timing of task executions and message exchanges. Compared to *DRP* (Section 4), *TTW*'s static schedules allow meeting shorter end-to-end deadlines (*Efficiency*) at the cost of a lesser *Adaptability*; indeed, *TTW*'s runtime adaptability is limited to switching between predefined operation modes.

6. Open issues and future work

Benchmarking Wireless Protocols. Our work on *TriScale* stemmed from discussions in the low-power wireless community regarding the need for a benchmark to compare networking protocols [56]. As we were reflecting on how to design such a benchmark, the need for a more rigorous experimental methodology became obvious; a need that *TriScale* tries to fill. We can now return to our initial objectives and attempt to realize the vision of IoTBench: a benchmark to thoroughly and confidently compare the performance of wireless networking protocols [115].
Going further with ST. With the design of *Baloo*, we attempted to make ST more accessible; an attempt that appears to be successful. Less than a year after the initial paper, the first independent studies using *Baloo* have been published [116]. There are many opportunities for future developments of the framework; the most natural being the port of *Baloo* to other platforms. It has been shown that the principle of ST also work on other physical layers that IEEE802.15.4 (*e.g.*, Bluetooth [25] and LoRa [26]). To investigate this further, a port to the LoRa-compatible SemTech SX1262 chip [117] is currently under development. A port to the Bluetooth-compatible nRF52840 Dongle [118] has just been recently released [119]. These would allow to experiment with ST-based networking on different physical layers and, by leveraging (hopefully upcoming) wireless protocol benchmarks, we would be able to objectively compare the performance trade-offs of these different technologies in a wide range of scenarios and applications.

On the application side, researchers are starting to harness the benefits of ST for wireless CPS, following the tracks opened by *TTW*—see *e.g.*, [92,120].
Dependable networking. One important limitation of the system designs presented in this dissertation is the reliance on a central authority, which we call *host*, in order to coordinate communication within the network. This creates a single point of failure: if the host should fail (or be jammed), the entire network would stop its operation. For any safety-critical applications, this is not acceptable. It is therefore important to work on system designs that would "distribute the responsibility" of the host. Recent contributions provide consensus primitives in low-power networks [25,116,121], an important piece for fault-tolerance in distributed systems. However, these works rely on a central authority for key network functions, such as time synchronization. More efforts are required to designed truly dependable wireless networks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] NSF, Cyber Physical Systems – Nsf10515, NSF, 2010, URL https://www.nsf.gov/pubs/2010/nsf10515/nsf10515.htm.
[2] J.A. Stankovic, When sensor and actuator networks cover the world, ETRI J. 30 (5) (2008) 627–633, http://dx.doi.org/10.4218/etrij.08.1308.0099, URL https://onlinelibrary.wiley.com/doi/abs/10.4218/etrij.08.1308.0099.
[3] E.A. Lee, Cyber physical systems: design challenges, in: 2008 11th IEEE International Symposium on Object and Component-Oriented Real-Time Distributed Computing, ISORC, 2008, pp. 363–369, http://dx.doi.org/10.1109/ISORC.2008.25.
[4] R. Rajkumar, I. Lee, L. Sha, J. Stankovic, Cyber-physical systems: the next computing revolution, in: Design Automation Conference, 2010, pp. 731–736, http://dx.doi.org/10.1145/1837274.1837461.
[5] M. Luvisotto, Z. Pang, D. Dzung, Ultra high performance wireless control for critical applications: challenges and directions, IEEE Trans. Ind. Inf. 13 (3) (2017) 1448–1459, http://dx.doi.org/10.1109/TII.2016.2617459.
[6] M. Haegele, Logistics drives 39% increase in professional service robot sales, IFR Int. Fed. Robot. (2018) URL https://ifr.org/post/logistics-drives-39-increase-in-professional-service-robot-sales.

[7] L. Mottola, M. Moretta, K. Whitehouse, C. Ghezzi, Team-level programming of drone sensor networks, in: Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems, in: SenSys '14, ACM, New York, NY, USA, 2014, pp. 177–190, http://dx.doi.org/10.1145/2668332.2668353, URL http://doi.acm.org/10.1145/2668332.2668353.

[8] J. Åkerberg, M. Gidlund, M. Björkman, Future research challenges in wireless sensor and actuator networks targeting industrial automation, in: 2011 9th IEEE International Conference on Industrial Informatics, 2011, pp. 410–415, http://dx.doi.org/10.1109/INDIN.2011.6034912.

[9] G.C. Buttazzo, Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications, Springer Science & Business Media, 2011.

[10] K. Geissdoerfer, B. Kusy, R. Jurdak, M. Zimmerling, Getting more out of energy-harvesting systems: energy management under time-varying utility with preact, in: 2019 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), 2019, pp. 109–120, http://dx.doi.org/10.1145/3302506.3310393.

[11] S. Weber, J. Beutel, R.D. Forno, A. Geiger, S. Gruber, T. Gsell, A. Hasler, M. Keller, R. Lim, P. Limpach, M. Meyer, I. Talzi, L. Thiele, C. Tschudin, A. Vieli, D. Vonder Mühll, M. Yücel, A decade of detailed observations (2008–2018) in steep bedrock permafrost at the matterhorn hörnligrat (zermatt, CH), Earth Syst. Sci. Data (2019) http://dx.doi.org/10.5194/essd-11-1203-2019, URL https://www.earth-syst-sci-data.net/11/1203/2019/.

[12] K. Chintalapudi, T. Fu, J. Paek, N. Kothari, S. Rangwala, J. Caffrey, R. Govindan, E. Johnson, S. Masri, Monitoring civil structures with a wireless sensor network, IEEE Internet Comput. 10 (2) (2006) 26–34, http://dx.doi.org/10.1109/MIC.2006.38.

[13] B. Cassens, M. Hartmann, T. Nowak, N. Duda, J. Thielecke, A. Kölpin, R. Kapitza, Bursting: increasing energy efficiency of erasure-coded data in animal-Borne sensor networks, in: Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks, in: EWSN '19, Junction Publishing, USA, 2019, pp. 59–70, URL http://dl.acm.org/citation.cfm?id=3324320.3324328.

[14] P. Zhang, C.M. Sadler, S.A. Lyon, M. Martonosi, Hardware design experiences in ZebraNet, in: Proceedings of the 2Nd International Conference on Embedded Networked Sensor Systems, in: SenSys '04, ACM, New York, NY, USA, 2004, pp. 227–238, http://dx.doi.org/10.1145/1031495.1031522, URL http://doi.acm.org/10.1145/1031495.1031522.

[15] Wikipedia, Chinese whispers, 2019, Wikipedia URL https://en.wikipedia.org/w/index.php?title=Chinese_whispers&oldid=920368233.

[16] T. Watteyne, V. Handziski, X. Vilajosana, S. Duquennoy, O. Hahm, E. Baccelli, A. Wolisz, Industrial wireless IP-based cyber-physical systems, Proc. IEEE 104 (5) (2016) 1025–1038, http://dx.doi.org/10.1109/JPROC.2015.2509186.

[17] H.-S. Kim, J. Ko, D.E. Culler, J. Paek, Challenging the IPv6 routing protocol for low-power and lossy networks (RPL): A survey, IEEE Commun. Surv. Tutor. 19 (4) (Fourthquarter 2017) 2502–2525, http://dx.doi.org/10.1109/COMST.2017.2751617.

[18] L. Mottola, G.P. Picco, MUSTER: adaptive energy-aware multisink routing in wireless sensor networks, IEEE Trans. Mob. Comput. 10 (12) (2011) 1694–1709, http://dx.doi.org/10.1109/TMC.2010.250.

[19] J. Stankovic, I. Lee, A. Mok, R. Rajkumar, Opportunities and Obligations for Physical Computing Systems, 2005, URL https://repository.upenn.edu/cis_papers/222.

[20] D. Yuan, M. Hollick, Let's talk together: understanding concurrent transmission in wireless sensor networks, in: 38th Annual IEEE Conference on Local Computer Networks, 2013, pp. 219–227, http://dx.doi.org/10.1109/LCN.2013.6761237.

[21] A. Escobar-Molero, Improving reliability and latency of wireless sensor networks using concurrent transmissions, At - Automatisierungstechnik 67 (1) (2019) 42–50, http://dx.doi.org/10.1515/auto-2018-0064, URL https://www.degruyter.com/view/j/auto.2019.67.issue-1/auto-2018-0064/auto-2018-0064.xml.

[22] M. Wilhelm, V. Lenders, J.B. Schmitt, On the reception of concurrent transmissions in wireless sensor networks, IEEE Trans. Wireless Commun. 13 (12) (2014) 6756–6767, http://dx.doi.org/10.1109/TWC.2014.2349896.

[23] F. Ferrari, M. Zimmerling, L. Thiele, O. Saukh, Efficient network flooding and time synchronization with glossy, in: Proceedings of the 10th ACM/IEEE International Conference on Information Processing in Sensor Networks, 2011, pp. 73–84, URL https://ieeexplore.ieee.org/document/5779066.

[24] D. Yuan, M. Riecker, M. Hollick, Making 'glossy' networks sparkle: exploiting concurrent transmissions for energy efficient, reliable, ultra-low latency communication in wireless control networks, in: Proceedings of the 11th European Conference on Wireless Sensor Networks - Vol. 8354, in: EWSN 2014, Springer-Verlag New York, Inc., New York, NY, USA, 2014, pp. 133–149, http://dx.doi.org/10.1007/978-3-319-04651-8_9.

[25] B. Al Nahas, S. Duquennoy, O. Landsiedel, Concurrent transmissions for multi-hop bluetooth 5, in: Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks, in: EWSN '19, Junction Publishing, USA, 2019, pp. 130–141, URL http://dl.acm.org/citation.cfm?id=3324320.3324336.

[26] M. Wegmann, Reliable 3rd Generation Data Collection, ETH Zurich, 2018, URL https://pub.tik.ee.ethz.ch/students/2018-FS/MA-2018-08.pdf.

[27] R. Lim, R. Da Forno, F. Sutton, L. Thiele, Competition: robust flooding using back-to-back synchronous transmissions with channel-hopping, in: Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks, in: EWSN '17, Junction Publishing, USA, 2017, pp. 270–271, URL http://dl.acm.org/citation.cfm?id=3108009.3108076.

[28] M. Baddeley, U. Raza, A. Stanoev, G. Oikonomou, R. Nejabati, M. Sooriyabandara, D. Simeonidou, Atomic-SDN: is synchronous flooding the solution to software-defined networking in IoT? IEEE Access 7 (2019) 96019–96034, http://dx.doi.org/10.1109/ACCESS.2019.2920100.

[29] X. Ma, P. Zhang, Y. Liu, X. Li, W. Tang, P. Tian, J. Wei, L. Shu, O. Theel, Competition: using decot+ to collect data under interference, in: Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks, 2019, p. 2, URL http://dl.acm.org/citation.cfm?id=3324320.3324385.

[30] F. Ferrari, M. Zimmerling, L. Mottola, L. Thiele, Low-power wireless bus, in: Proceedings of the 10th ACM Conference on Embedded Network Sensor Systems, in: SenSys '12, ACM, New York, NY, USA, 2012, pp. 1–14, http://dx.doi.org/10.1145/2426656.2426658, URL http://doi.acm.org/10.1145/2426656.2426658.

[31] M. Zimmerling, L. Mottola, S. Santini, Synchronous transmissions in low-power wireless: a survey of communication protocols and network services, ACM Comput. Surv. 53 (6) (2020) 121:1–121:39, http://dx.doi.org/10.1145/3410159.

[32] O. Landsiedel, F. Ferrari, M. Zimmerling, Chaos: versatile and efficient all-to-all data sharing and in-network processing at scale, in: Proceedings of the 11th ACM Conference on Embedded Networked Sensor Systems, in: SenSys '13, ACM, New York, NY, USA, 2013, pp. 1:1–1:14, http://dx.doi.org/10.1145/2517351.2517358, URL http://doi.acm.org/10.1145/2517351.2517358.

[33] C. Herrmann, F. Mager, M. Zimmerling, Mixer: efficient many-to-all broadcast in dynamic wireless mesh networks, in: Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems, in: SenSys '18, ACM, New York, NY, USA, 2018, pp. 145–158, http://dx.doi.org/10.1145/3274783.3274849, URL http://doi.acm.org/10.1145/3274783.3274849.

[34] M. Schuß, C.A. Boano, M. Weber, K. Römer, A competition to push the dependability of low-power wireless protocols to the edge, in: Proceedings of the 14th International Conference on Embedded Wireless Systems and Networks, in: EWSN'17, Junction Publishing, USA, 2017, pp. 54–65, http://dx.doi.org/10.5555/3108009.3108018.

[35] A. Escobar, F. Moreno, A.J. Cabrera, J. Garcia-Jimenez, F.J. Cruz, U. Ruiz, J. Klaue, A. Corona, D. Tati, T. Meyerhoff, Competition: BigBangBus, in: Proceedings of the 2018 International Conference on Embedded Wireless Systems and Networks, in: EWSN '18, Junction Publishing, USA, 2018, pp. 213–214, URL http://dl.acm.org/citation.cfm?id=3234847.3234894.

[36] P. Sommer, Y.-A. Pignolet, Competition: dependable network flooding using glossy with channel-hopping, in: Proceedings of the 2016 International Conference on Embedded Wireless Systems and Networks, 2016, URL http://dl.acm.org/citation.cfm?id=2893711.2893785.

[37] A. Escobar-Molero, J. Garcia-Jimenez, J. Klaue, F. Moreno-Cruz, B. Saez, F.J. Cruz, U. Ruiz, A. Corona, Competition: RedNodeBus, stretching out the preamble, in: Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks, in: EWSN '19, Junction Publishing, USA, 2019, pp. 304–305, URL http://dl.acm.org/citation.cfm?id=3324320.3324391.

[38] NXP, LPC541XX, URL https://www.nxp.com/products/processors-and-microcontrollers/arm-microcontrollers/general-purpose-mcus/lpc54000-cortex-m4-/low-power-microcontrollers-mcus-based-on-arm-cortex-m4-cores-with-optional-cortex-m0-plus-co-processor:LPC541XX.

[39] NXP, VF3xxR URL https://www.nxp.com/products/processors-and-microcontrollers/legacy-mcu-mpus/vfxxx-controller/r-series/32-bit-devices-for-advanced-connected-radio-entry-level-infotainment-and-digital-instrument-cluster-applications.:VF3xxR.

[40] J. Beutel, R. Trüb, R.D. Forno, M. Wegmann, T. Gsell, R. Jacob, M. Keller, F. Sutton, L. Thiele, The dual processor platform architecture: demo abstract, in: Proceedings of the 18th International Conference on Information Processing in Sensor Networks, in: IPSN '19, ACM, New York, NY, USA, 2019, pp. 335–336, http://dx.doi.org/10.1145/3302506.3312481, URL http://doi.acm.org/10.1145/3302506.3312481.

[41] F. Sutton, M. Zimmerling, R. Da Forno, R. Lim, T. Gsell, G. Giannopoulou, F. Ferrari, J. Beutel, L. Thiele, Bolt: a stateful processor interconnect, in: Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, in: SenSys '15, ACM, New York, NY, USA, 2015, pp. 267–280, http://dx.doi.org/10.1145/2809695.2809706, URL http://doi.acm.org/10.1145/2809695.2809706.

[42] Texas Instruments, MSP-EXP432P401R SimpleLink™, URL http://www.ti.com/tool/MSP-EXP432P401R.

[43] Texas Instruments, CC430F6137 16-bit ultra-low-power MCU, 2019, [Online] URL http://www.ti.com/product/CC430F6137. (Accessed 11 January 2019).

[44] S. Duquennoy, B. Al Nahas, O. Landsiedel, T. Watteyne, Orchestra: robust mesh networks through autonomously scheduled TSCH, in: Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, in: SenSys '15, ACM, New York, NY, USA, 2015, pp. 337–350, http://dx.doi.org/10.1145/2809695.2809714, URL http://doi.acm.org/10.1145/2809695.2809714.

[45] S. Duquennoy, J. Eriksson, T. Voigt, Five-nines reliable downward routing in RPL, 2017, Cs arXiv:1710.02324.

[46] T. Istomin, M. Trobinger, A.L. Murphy, G.P. Picco, Interference-resilient ultra-low power aperiodic data collection, in: Proceedings of the 17th ACM/IEEE International Conference on Information Processing in Sensor Networks, in: IPSN '18, IEEE Press, Piscataway, NJ, USA, 2018, pp. 84–95, http://dx.doi.org/10.1109/IPSN.2018.00015.

[47] M. Matos, Towards reproducible evaluation of large-scale distributed systems, in: Proceedings of the International Workshop on Advanced Tools, Programming Languages, and Platforms for Implementing and Evaluating Algorithms for Distributed Systems, ApPLIED, ACM, Egham, United Kingdom, 2018, pp. 5–7, http://dx.doi.org/10.1145/3231104.3231113, URL http://doi.acm.org/10.1145/3231104.3231113.

[48] R. Burchfield, E. Nourbakhsh, J. Dix, K. Sahu, S. Venkatesan, R. Prakash, RF in the jungle: effect of environment assumptions on wireless experiment repeatability, in: Proceedings of the International Conference on Communications, (ICC), IEEE, 2009, pp. 1–6, URL https://ehsaan.net/wp-content/uploads/publications/rfij.pdf.

[49] A. Maricq, D. Duplyakin, I. Jimenez, C. Maltzahn, R. Stutsman, R. Ricci, Taming performance variability, in: Proceedings of the 13th International USENIX Symposium on Operating Systems Design and Implementation, OSDI, USENIX Association, Carlsbad, CA, USA, 2018, pp. 409–425, URL https://www.usenix.org/system/files/osdi18-maricq.pdf.

[50] S.M. Blackburn, A. Diwan, M. Hauswirth, P.F. Sweeney, J.N. Amaral, T. Brecht, L. Bulej, C. Click, L. Eeckhout, S. Fischmeister, D. Frampton, L.J. Hendren, M. Hind, A.L. Hosking, R.E. Jones, T. Kalibera, N. Keynes, N. Nystrom, A. Zeller, The truth, the whole truth, and nothing but the truth: a pragmatic guide to assessing empirical evaluations, ACM Trans. Program. Lang. Syst. 38 (4) (2016) 15:1–15:20, http://dx.doi.org/10.1145/2983574.

[51] V. Bajpai, O. Bonaventure, K. Claffy, D. Karrenberg, Encouraging reproducibility in scientific research of the internet (dagstuhl seminar 18412), in: V. Bajpai, O. Bonaventure, K. Claffy, D. Karrenberg (Eds.), Dagstuhl Rep. 8 (10) (2019) 41–62, http://dx.doi.org/10.4230/DagRep.8.10.41, URL http://drops.dagstuhl.de/opus/volltexte/2019/10347.

[52] M. Baker, Is there a reproducibility crisis? Nat. News 533 (7604) (2016) 452–454, URL https://www.nature.com/news/polopoly_fs/1.19970!/menu/main/topColumns/topLeftColumn/pdf/533452a.pdf.

[53] C. Collberg, T. Proebsting, A.M. Warren, Repeatability and Benefaction in Computer Systems Research, Technical Report TR 14–04, University of Arizona, 2015, URL http://reproducibility.cs.arizona.edu/v2/RepeatabilityTR.pdf.

[54] D. Saucez, L. Iannone, Thoughts and recommendations from the ACM SIGCOMM 2017 reproducibility workshop, SIGCOMM Comput. Commun. Rev. 48 (1) (2018) 70–74, http://dx.doi.org/10.1145/3211852.3211863, URL https://ccronline.sigcomm.org/wp-content/uploads/2017/11/sigcomm-ccr-paper149.pdf.

[55] V. Bajpai, A. Brunstrom, A. Feldmann, W. Kellerer, A. Pras, H. Schulzrinne, G. Smaragdakis, M. Wählisch, K. Wehrle, The dagstuhl beginners guide to reproducibility for experimental networking research, SIGCOMM Comput. Commun. Rev. 49 (1) (2019) 24–30, URL https://dl.acm.org/citation.cfm?id=3314217.

[56] C.A. Boano, S. Duquennoy, A. Förster, O. Gnawali, R. Jacob, H.-S. Kim, O. Landsiedel, R. Marfievici, L. Mottola, G.P. Picco, X. Vilajosana, T. Watteyne, M. Zimmerling, IoTBench: towards a benchmark for low-power wireless networking, in: Proceedings of the 1st Workshop on Benchmarking Cyber-Physical Networks and Systems (CPSBench 2018), 2018, http://dx.doi.org/10.3929/ethz-b-000256517, URL https://www.research-collection.ethz.ch/handle/20.500.11850/256517.

[57] R. Jacob, M. Zimmerling, C.A. Boano, L. Vanbever, L. Thiele, Designing replicable networking experiments with triscale, J. Syst. Res. 1 (1) (2021) http://dx.doi.org/10.5070/SR31155408, URL https://escholarship.org/uc/item/63n4s9w2.

[58] R. Jacob, J. Bächli, R. Da Forno, L. Thiele, Synchronous transmissions made easy: design your network stack with baloo, in: Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks, 2019, http://dx.doi.org/10.3929/ethz-b-000324254.

[59] R. Jacob, M. Zimmerling, P. Huang, J. Beutel, L. Thiele, End-to-end real-time guarantees in wireless cyber-physical systems, in: 2016 IEEE Real-Time Systems Symposium, RTSS, 2016, pp. 167–178, http://dx.doi.org/10.1109/RTSS.2016.025, URL https://www.research-collection.ethz.ch/handle/20.500.11850/221542.

[60] R. Jacob, L. Zhang, M. Zimmerling, J. Beutel, S. Chakraborty, L. Thiele, The time-triggered wireless architecture, in: M. Völp (Ed.), 32nd Euromicro Conference on Real-Time Systems, ECRTS 2020, in: Leibniz International Proceedings in Informatics (LIPIcs), 165, Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 2020, pp. 19:1–19:25, http://dx.doi.org/10.4230/LIPIcs.ECRTS.2020.19.

[61] H.E. Plesser, Reproducibility vs. replicability: a brief history of a confused terminology, Front. Neuroinform. 11 (76) (2018) 1–4.

[62] L.A. Barba, Terminologies for reproducible research, 2018, arXiv:1802.03311 [Cs] arXiv:1802.03311.

[63] ACM, Artifact review and badging, 2020, URL https://www.acm.org/publications/policies/artifact-review-badging.

[64] V. Paxson, Strategies for sound internet measurement, in: Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement, in: IMC '04, Association for Computing Machinery, Taormina, Sicily, Italy, 2004, pp. 263–271, http://dx.doi.org/10.1145/1028788.1028824.

[65] J. Vitek, T. Kalibera, Repeatability, reproducibility and rigor in systems research, in: Proceedings of the 9th International Conference on Embedded Software (EMSOFT), ACM, 2011, pp. 33–38, URL https://www.cs.kent.ac.uk/pubs/2011/3174/content.pdf.

[66] K. Kritsis, G.Z. Papadopoulos, A. Gallais, P. Chatzimisios, F. Théoleyre, A tutorial on performance evaluation and validation methodology for low-power and lossy networks, IEEE Commun. Surv. Tutor. (2018) 1, http://dx.doi.org/10.1109/COMST.2018.2820810, URL https://icube-publis.unistra.fr/docs/13111/IEEECOMST-Perf-LLN.pdf.

[67] M. Flittner, M.N. Mahfoudi, D. Saucez, M. Wählisch, L. Iannone, V. Bajpai, A. Afanasyev, A survey on artifacts from CoNEXT, ICN, IMC, and SIGCOMM conferences in 2017, SIGCOMM Comput. Commun. Rev. 48 (1) (2018) 75–80, http://dx.doi.org/10.1145/3211852.3211864, URL http://doi.acm.org/10.1145/3211852.3211864.

[68] L. Nussbaum, Testbeds support for reproducible research, in: Proceedings of the International ACM SIGCOMM Reproducibility Workshop, in: Reproducibility'17, ACM, Los Angeles, CA, USA, 2017, pp. 24–26, http://dx.doi.org/10.1145/3097766.3097773, URL https://hal.inria.fr/hal-01577849/document.

[69] F.Y. Yan, J. Ma, G.D. Hill, D. Raghavan, R.S. Wahby, P. Levis, K. Winstein, Pantheon: the training ground for internet congestion-control research, in: Proceedings of the International USENIX Annual Technical Conference, ATC, USENIX Association, Boston, MA, USA, 2018, pp. 731–743, URL https://www.usenix.org/conference/atc18/presentation/yan-francis.

[70] F.Y. Yan, H. Ayers, C. Zhu, S. Fouladi, J. Hong, K. Zhang, P. Levis, K. Winstein, Learning in situ: A randomized experiment in video streaming, in: 17th USENIX Symposium on Networked Systems Design and Implementation, NSDI 20, 2020, pp. 495–511, URL https://www.usenix.org/conference/nsdi20/presentation/yan.

[71] A. Uta, A. Custura, D. Duplyakin, I. Jimenez, J. Rellermeyer, C. Maltzahn, R. Ricci, A. Iosup, Is big data performance reproducible in modern cloud networks? in: Proceedings of the 17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20), 2020, pp. 513–527, URL https://www.usenix.org/conference/nsdi20/presentation/uta.

[72] M. Focus, Seven Ways to Fail, Technical Report Brochure on Application Development, Test, and Delivery, Micro Focus, 2018, URL https://www.microfocus.com/media/brochure/seven_ways_to_fail_brochure.pdf.

[73] R. Jacob, C.A. Boano, U. Raza, M. Zimmerling, L. Thiele, Towards a methodology for experimental evaluation in low-power wireless networking, in: Proceedings of the 2nd Workshop on Benchmarking Cyber-Physical Systems and Internet of Things, CPS-IoTBench'19, 2019, http://dx.doi.org/10.3929/ethz-b-000325096, URL https://www.research-collection.ethz.ch/handle/20.500.11850/325096.

[74] R. Jacob, TriScale, 2021, URL https://github.com/romain-jacob/triscale.

[75] R. Jacob, TriScale Demo, 2021, URL http://triscale.ethz.ch.

[76] A. Escobar, F.J. Cruz, J. Garcia-Jimenez, J. Klaue, A. Corona, RedFixHop with channel hopping: reliable ultra-low-latency network flooding, in: 2016 Conference on Design of Circuits and Integrated Systems, (DCIS), 2016, pp. 1–4, http://dx.doi.org/10.1109/DCIS.2016.7845367.

[77] B. Al Nahas, S. Duquennoy, O. Landsiedel, Network-wide consensus utilizing the capture effect in low-power wireless networks, in: Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems, in: SenSys '17, ACM, New York, NY, USA, 2017, pp. 1:1–1:14, http://dx.doi.org/10.1145/3131672.3131685, URL http://doi.acm.org/10.1145/3131672.3131685.

[78] M. Mohammad, M.C. Chan, Codecast: supporting data driven in-network processing for low-power wireless sensor networks, in: Proceedings of the 17th ACM/IEEE International Conference on Information Processing in Sensor Networks, in: IPSN '18, IEEE Press, Piscataway, NJ, USA, 2018, pp. 72–83, http://dx.doi.org/10.1109/IPSN.2018.00014.

[79] W. Du, J.C. Liando, H. Zhang, M. Li, When pipelines meet fountain: fast data dissemination in wireless sensor networks, in: Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, in: SenSys '15, ACM, New York, NY, USA, 2015, pp. 365–378, http://dx.doi.org/10.1145/2809695.2809721, URL http://doi.acm.org/10.1145/2809695.2809721.

[80] M. Mohammad, M. Doddavenkatappa, M.C. Chan, Improving performance of synchronous transmission-based protocols using capture effect over multichannels, ACM Trans. Sen. Netw. (2017) http://dx.doi.org/10.1145/3043790, URL http://doi.acm.org/10.1145/3043790.

[81] P. Zhang, A.Y. Gao, O. Theel, Less is more: learning more with concurrent transmissions for energy-efficient flooding, in: Proceedings of the 14th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services, in: MobiQuitous 2017, ACM, New York, NY, USA, 2017, pp. 323–332, http://dx.doi.org/10.1145/3144457.3144482.

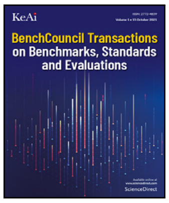
[82] F. Sutton, R. Da Forno, D. Gschwend, T. Gsell, R. Lim, J. Beutel, L. Thiele, The design of a responsive and energy-efficient event-triggered wireless sensing system, in: Proceedings of the 2017 International Conference on Embedded Wireless Systems and Networks, in: EWSN '17, Junction Publishing, USA, 2017, pp. 144–155, URL http://dl.acm.org/citation.cfm?id=3108009.3108028.

[83] C. Sarkar, R.V. Prasad, R.T. Rajan, K. Langendoen, Sleeping beauty: efficient communication for node scheduling, in: 2016 IEEE 13th International Conference on Mobile Ad Hoc and Sensor Systems (MASS), 2016, pp. 56–64, http://dx.doi.org/10.1109/MASS.2016.018.
[84] L. Mottola, G.P. Picco, Middleware for wireless sensor networks: An outlook, J. Internet Serv. Appl. 3 (1) (2012) 31–39, http://dx.doi.org/10.1007/s13174-011-0046-7, URL https://link.springer.com/article/10.1007/s13174-011-0046-7.
[85] Advanticsys, MTM-CM5000-MSP 802.15.4 telosb mote module, 2019, [Online] URL https://www.advanticsys.com/shop/mtmcm5000msp-p-14.html. (Accessed 11 January 2019).
[86] V. Poirot, O. Landsiedel, Dimmer: self-adaptive network-wide flooding with reinforcement learning, in: 2021 IEEE 41st International Conference on Distributed Computing Systems, ICDCS, 2021, pp. 293–303, http://dx.doi.org/10.1109/ICDCS51616.2021.00036.
[87] M. Baddeley, A. Aijaz, U. Raza, A. Stanoev, Y. Jin, M. Schuß, C.A. Boano, G. Oikonomou, 6TiSCH++ with bluetooth 5 and concurrent transmissions, 2020, Cs arXiv:2010.09529.
[88] B.-M. Cho, S. Kim, K.-D. Kim, K.-J. Park, A controller switching mechanism for resilient wireless sensor–actuator networks, Appl. Sci. 12 (4) (2022) 1841, http://dx.doi.org/10.3390/app12041841, URL https://www.mdpi.com/2076-3417/12/4/1841.
[89] J. Oostvogels, F. Yang, S. Michiels, D. Hughes, Zero-wire: A deterministic and low-latency wireless bus through symbol-synchronous transmission of optical signals, in: Proceedings of the 18th Conference on Embedded Networked Sensor Systems, Association for Computing Machinery, New York, NY, USA, 2020, pp. 164–178.
[90] X. Ma, P. Zhang, O. Theel, J. Wei, Gathering data with packet-in-packet in wireless sensor networks, Comput. Netw. 170 (2020) 107124, http://dx.doi.org/10.1016/j.comnet.2020.107124, URL https://www.sciencedirect.com/science/article/pii/S1389128619306255.
[91] C. Sarkar, R.V. Prasad, K. Langendoen, FLEET: when time-bounded communication meets high energy-efficiency, IEEE Access 7 (2019) 77555–77568, http://dx.doi.org/10.1109/ACCESS.2019.2920937.
[92] F. Mager, D. Baumann, C. Herrmann, S. Trimpe, M. Zimmerling, Scaling beyond bandwidth limitations: wireless control with stability guarantees under overload, ACM Trans. Cyber-Phys. Syst. (2021) http://dx.doi.org/10.1145/3502299.
[93] R. Jacob, Baloo, 2018, [Online] URL http://www.romainjacob.net/baloo/ (Accessed 11 January 2019).
[94] C. Lu, B. Blum, T. Abdelzaher, J. Stankovic, T. He, RAP: A real-time communication architecture for large-scale wireless sensor networks, in: Proceedings. Eighth IEEE Real-Time and Embedded Technology and Applications Symposium, 2002, pp. 55–66, http://dx.doi.org/10.1109/RTTAS.2002.1137381.
[95] J.A. Stankovic, T.F. Abdelzaher, L. Chenyang, S. Lui, J.C. Hou, Real-time communication and coordination in embedded sensor networks, Proc. IEEE 91 (7) (2003) 1002–1022, http://dx.doi.org/10.1109/JPROC.2003.814620.
[96] T. He, J.A. Stankovic, C. Lu, T. Abdelzaher, SPEED: A stateless protocol for real-time communication in sensor networks, in: 23rd International Conference on Distributed Computing Systems, 2003. Proceedings, 2003, pp. 46–55, http://dx.doi.org/10.1109/ICDCS.2003.1203451.
[97] International Electrotechnical Commission (IEC), Industrial networks - wireless communication network and communication profiles - wirelesshart, 2020, URL https://webstore.iec.ch/publication/24433. (Accessed 14 April 2020).
[98] ISA100, Wireless compliance institute, 2009, URL http://www.isa100wci.org/.
[99] W. Watteyne, P. Tuset-Peiro, X. Vilajosana, S. Pollin, B. Krishnamachari, Teaching communication technologies and standards for the industrial IoT? use 6tisch!, IEEE Commun. Mag. 55 (5) (2017) 132–137, http://dx.doi.org/10.1109/MCOM.2017.1700013.
[100] A. Saifullah, Y. Xu, C. Lu, Y. Chen, Real-time scheduling for wirelesshart networks, in: 2010 31st IEEE Real-Time Systems Symposium, 2010, pp. 150–159, http://dx.doi.org/10.1109/RTSS.2010.41.
[101] A. Saifullah, Y. Xu, C. Lu, Y. Chen, End-to-end communication delay analysis in industrial wireless networks, IEEE Trans. Comput. 64 (5) (2015) 1361–1374, http://dx.doi.org/10.1109/TC.2014.2322609.
[102] M. Zimmerling, L. Mottola, P. Kumar, F. Ferrari, L. Thiele, Adaptive real-time communication for wireless cyber-physical systems, ACM Trans. Cyber-Phys. Syst. 1 (2) (2017) 8:1–8:29, http://dx.doi.org/10.1145/3012005.
[103] R. Jacob, Distributed real-time protocol – DRP, 2019, URL https://github.com/romain-jacob/drp.
[104] T. Abdelzaher, K. Shin, Combined task and message scheduling in distributed real-time systems, IEEE Trans. Parallel Distrib. Syst. 10 (11) (1999) 1179–1191, http://dx.doi.org/10.1109/71.809575.
[105] S.S. Craciunas, R.S. Oliver, Combined task- and network-level scheduling for distributed time-triggered systems, Real-Time Syst. 52 (2) (2016) 161–200, http://dx.doi.org/10.1007/s11241-015-9244-x, URL http://dx.doi.org/10.1007/s11241-015-9244-x.
[106] M. Ashjaei, N. Khalilzad, S. Mubeen, M. Behnam, I. Sander, L. Almeida, T. Nolte, Designing end-to-end resource reservations in predictable distributed embedded systems, Real-Time Syst. 53 (6) (2017) 916–956, http://dx.doi.org/10.1007/s11241-017-9283-6.
[107] G. Fohler, Changing operational modes in the context of pre run-time scheduling, IEICE Trans. Inf. Syst. 76 (11) (1993) 1333–1340, URL https://pdfs.semanticscholar.org/272b/615266e763369e903dcb0b966e22077f127c.pdf.
[108] Wikipedia, Bin packing problem, 2019, Wikipedia URL https://en.wikipedia.org/w/index.php?title=Bin_packing_problem&oldid=921684516.
[109] A. Azim, Scheduling of Overload-Tolerant Computation and Multi-Mode Communication in Real-Time Systems (Ph.D. thesis), University of Waterloo, 2014, URL http://hdl.handle.net/10012/8973.
[110] W. Steiner, An evaluation of SMT-based schedule synthesis for time-triggered multi-hop networks, in: 2010 31st IEEE Real-Time Systems Symposium, 2010, pp. 375–384, http://dx.doi.org/10.1109/RTSS.2010.25.
[111] S.S. Craciunas, R.S. Oliver, SMT-based task- and network-level static schedule generation for time-triggered networked systems, in: Proceedings of the 22Nd International Conference on Real-Time Networks and Systems, in: RTNS '14, ACM, New York, NY, USA, 2014, pp. 45:45–45:54, http://dx.doi.org/10.1145/2659787.2659812, URL http://doi.acm.org/10.1145/2659787.2659812.
[112] J. Huang, J.O. Blech, A. Raabe, C. Buckl, A. Knoll, Static scheduling of a time-triggered network-on-chip based on SMT solving, in: 2012 Design, Automation Test in Europe Conference Exhibition (DATE), 2012, pp. 509–514, http://dx.doi.org/10.1109/DATE.2012.6176522.
[113] K. Jeffay, D.F. Stanat, C.U. Martel, On non-preemptive scheduling of period and sporadic tasks, in: Proceedings Twelfth Real-Time Systems Symposium, 1991, pp. 129–139, http://dx.doi.org/10.1109/REAL.1991.160366.
[114] R. Jacob, The Time-Triggered Wireless Architecture URL https://ttw.ethz.ch/.
[115] IoT Benchmarks Initiative, IoTBench URL https://www.iotbench.ethz.ch/.
[116] A. Spina, M. Breza, N. Dulay, J. McCann, XPC: fast and reliable synchronous transmission protocols for 2-phase commit and 3-phase commit, 2019, Cs arXiv:1910.09941.
[117] Semtech, SX1262. Long Range Low Power LoRa URL https://www.semtech.com/products/wireless-rf/lora-transceivers/sx1262.
[118] Nordic Semiconductors, Nrf52840, 2018, [Online] URL https://www.nordicsemi.com/Products/Low-power-short-range-wireless/nRF52840. (Accessed 16 May 2020).
[119] L. Tuchtenhagen, More Easy Synchronous Transmissions: Expanding Baloo to the nRF52840 (Ph.D. thesis), 2022, URL https://www.ds.informatik.uni-kiel.de/en/teaching/bachelor-and-master-theses/completed-master-and-bachelor-theses/2021_Bachelor-Thesis-Lars-Tuchtenhagen.pdf.
[120] D. Baumann, F. Mager, U. Wetzker, L. Thiele, M. Zimmerling, S. Trimpe, Wireless control for smart manufacturing: recent approaches and open challenges, Proc. IEEE 109 (4) (2021) 441–467, http://dx.doi.org/10.1109/JPROC.2020.3032633.
[121] V. Poirot, B. Al Nahas, O. Landsiedel, Paxos made wireless: consensus in the air, in: Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks, in: EWSN '19, Junction Publishing, USA, 2019, pp. 1–12, URL http://dl.acm.org/citation.cfm?id=3324320.3324322.

Contents lists available at ScienceDirect

BenchCouncil Transactions on Benchmarks, Standards and Evaluations

journal homepage: https://www.keaipublishing.com/en/journals/benchcouncil-transactions-on-benchmarks-standards-and-evaluations/



Training, testing and benchmarking medical AI models using Clinical AIBench

Yunyou Huang [a,f], Xiuxia Miao [a], Ruchang Zhang [a], Li Ma [c], Wenjing Liu [a], Fan Zhang [b], Xianglong Guan [a], Xiaoshuang Liang [a], Xiangjiang Lu [a], Suqing Tang [e], Zhifei Zhang [d,*]

[a] *Guangxi Key Lab of Multi-Source Information Mining & Security, School of Computer Science and Engineering & School of Software, Guangxi Normal University, Guilin, China*
[b] *State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China*
[c] *Guilin Medical University, Guilin, China*
[d] *Department of Physiology and Pathophysiology, Capital Medical University, Beijing, China*
[e] *Faculty of Education, Guangxi Normal University, Guilin, China*
[f] *International Open Benchmark Council, Beijing, China*

ARTICLE INFO

Keywords:
Benchmark
Clinical setting
Alzheimer's disease
COVID-19
Dental
Configurable clinical setting

ABSTRACT

AI technology has been used in many clinical research fields, but most AI technologies are difficult to land in real-world clinical settings. In most current clinical AI research settings, the diagnosis task is to identify different types of diseases among the given ones. However, the diagnosis in real-world settings needs dynamically developing inspection strategies based on the existing resources of medical institutions and identifying different kinds of diseases out of many possibilities. To promote the development of different clinical AI technologies and the implementation of clinical applications, we propose a benchmark named Clinical AIBench for developing, verifying, and evaluating clinical AI technologies in real-world clinical settings. Specifically, Clinical AIBench can be used for: (1) Model training and testing: Researchers can use the data to train and test their models. (2)Model evaluation: Researchers can use Clinical AIBench to objectively, fairly, and comparably evaluate various models of different researchers. (3) Clinical value evaluation: Researchers can use the clinical indicators provided by Clinical AIBench to evaluate the clinical value of models, which will be applied in real-world clinical settings. For convenience, Clinical AIBench provides three different levels of clinical settings: restricted clinical setting, which is named closed clinical setting, data island clinical setting, and real-world clinical setting, which is called open clinical setting. In addition, Clinical AIBench covers three diseases: Alzheimer's disease, COVID-19, and dental. Clinical AIBench provides python APIs to researchers. The data and source code are publicly available from the project website https://www.benchcouncil.org/clinical_aibench/.

1. Introduction

Data is the cornerstone of current AI technologies research, which has become the consensus of researchers. From ImageNet [1] and MNIST [2] in the general field to ADNI [3] and TCGA [4] in the clinical field, the datasets have greatly promoted the development of AI technologies research. However, many current AI technologies proposed by research work are difficult to land in the real world. Especially in clinical medicine, only 71 clinical AI technologies have been approved by the FDA by now [5]. An AI-assisted decision-making device has been used to detect the fetus's heart rate during childbirth in pregnant women. Still, the practice proved that the device could not improve the clinical outcome of the mother or baby; Instead, it increases the workload of clinicians [6].

Why the AI technology loses its power in the real world? The answer is that the essentials of the same clinical tasks are changed in different clinical settings. For example, the diagnosis tasks in most current clinical AI research settings are identifying different types of diseases among the given diseases. The diagnosis in the isolated data island clinical setting, also called federated learning setting [7], has three phases: first, ensure data security; second, use the data from different institutions to train a model; finally, identify different types of diseases among the given ones. Constantly, the diagnosis in real-world settings, which we also call open clinical settings, needs dynamically developing inspection strategies based on the existing resources of medical institutions, locating subjects in one type or several types of diseases, and identifying different kinds of diseases. Thus, solving a task in a clinical setting, e.g., a closed or data island clinical setting, does not mean it is well done in an open clinical setting.

To promote the clinical AI technology implementation, we propose a scenario-based benchmark suite [8], named Clinical AIBench, which

* Corresponding author.
E-mail address: zhifeiz@ccmu.edu.cn (Z. Zhang).

https://doi.org/10.1016/j.tbench.2022.100037
Received 7 March 2022; Received in revised form 24 March 2022; Accepted 24 March 2022
Available online 12 April 2022

is a measurement standard and also requires rigorous validation [9], to develop, verify, and evaluate the clinical AI technologies and systems [10]. In detail, Clinical AIBench can be used for (1) model training and testing. For example, the closed clinical setting mentioned in Section 5.1, the data island clinical setting mentioned in Section 5.2, and the open clinical setting mentioned in Section 5.3 can be used for training and testing Alzheimer's disease diagnosis models under different requirements. (2) Model evaluation. For example, the open clinical setting mentioned in Section 5.3 reproduces the clinical setting's main features of the real Alzheimer's disease diagnosis, not allowed to be modified. Not only does it can be used to train and test Alzheimer's disease diagnosis model, but it also can be used as a fair, objective, and comparable tool to evaluate various models from different researchers. (3) Clinical value evaluation. For example, the **get_damage_level** and **get_cost** APIs mentioned in Section 4 can be used to evaluate the cost and the degree of damage of the subjects during the diagnosis. The risks indicators and the subject's benefits during diagnosis are an integrated part of Clinical AIBench.

2. The installation of Clinical AIBench

Clinical AIBench is an open-source tool and provides services according to the Client-Services model (currently, Clinical AIBench only runs on Linux and Mac systems). Clinical AIBench is designed to be similar to the FLBench [7] structure, which has four parts: input data, scenario configuration, scenario, and automated deployment tools. The researcher can download the clients of Clinical AIBench from this website. Due to the particularity of clinical data and the requirements of the organization of data owners, researchers must register and follow all relevant protocols.

As shown in Fig. 1, Clinical AIBench requires 13 packages: configparser, pandas, numpy, nibabel, flask, scikit-learn, requests, imageio, joblib, tensorflow, torch, wandb, fedml [11] etc.

3. The configuration of clinical setting

In Clinical AIBench, the clinical setting is constructed according to the configuration file. For every typical clinical setting, Clinical AIBench will provide a configuration file to describe the characteristic of the clinical setting and build the clinical setting in Clinical AIBench. In addition, the researcher is able to upload the configuration file to construct their clinical setting according to their needs. As shown in Fig. 2, the format of the configuration file used in Clinical AIBench is *ini*. The *ini* file has three elements: section, parameter, and comment. The *section* is enclosed in a square brackets(*[]*), and an *ini* file contains one or more section. The *parameter* is a key–value pair. All parameters after the section declaration belong to the section. The *comment* starts with *;* or *#*, which means the whole line is a comment. As shown in Fig. 3 is a specific example of the use of *ini* file.

As shown in Fig. 3, this configuration file has two sections. Each section contains multiple parameters. The first section, named *scenario*, describes some basic information about the *scenario*, such as the *scenario* name, the *scenario* version number, etc. The second section, named *block1*, describes the data information used in the *scenario*. The meaning of parameters of the *block1* will be described in detail below.

(1) *files* specifies the data files that need to be used in the scenario. The requirements for the data file are as follows: (1) The file type must be *csv* format. (2) The first line of the file must be the column names. (3) Special symbols cannot exist in column names, such as @, ., & etc. (4) The path of image, audio, or video will be filled in the file if the data file contains the image, audio, or video resource.

(2) *joinorder* specifies the order that connect data files, such as *joinorder = file1, file2.*

```
configparser==5.0.2
pandas==1.0.5
numpy==1.18.5
nibabel==3.2.1
flask==1.1.2
scikit-learn==0.23.1
requests==2.24.0
imageio==2.9.0
joblib==1.0.1
tensorflow==2.6.0rc0
torch==1.9.1
wandb==0.12.2
fedml==0.1
```

Fig. 1. Packages.

```
[section1]
key1=value1, value2, ...
key2=value1, value2, ...

[section2]
key1=value1, value2, ...
key2=value1, value2, ...

```

Fig. 2. Ini format.

(3) *connection* specifies the primary keys that connect different data files, such as *connection = file1|column1&file2|column1+ file1 |column2&file2|column2.*

(4) *columns* specifies the column that selected in the file according to column name, such as *columns = files1|column1, files1|column2, files2|column1, files2|column2.* And all data in the file is selected if the column name is no designated.

(5) *rows* specifies the row selection rule of the file according to the specified conditions, such as *rows = file1|column1 = value1, file1|column2! = value2.* And all data in the file is selected if the *row* is no designated.

(6) *label* specifies which column of the file as the label of machine learning task, such as *label = file| column1.*

(7) *partition_number* specifies the number of clients. If the number is 1, which is a closed clinical setting. If the number is greater than 1, which is a data island clinical setting with multiple clients. If the number is −1, which is a data island clinical setting with an indeterminate number of clients.

(8) *assignment* specifies the method that divide datasets in the data island clinical setting: (1) *_scope* the function of this method is divide datasets according to sorted of the datasets and the parameter *rate* in the configuration file; (2) *_class* the function of this method is divide the datasets according to the type of value that is designated column; (3) *_constraint* the function of this method is divide the datasets according to the parameter *condition* in the configuration file.

```
[scenarion]

name
version
author

[block1]

files

joinorder

connection

columns

rows

label

partition_number

;assignment=scope()
;assignment=class()
;assignment=constraint()

;rate

;conditions

;add

image

audio

video
```

Fig. 3. Configuration file.

(9) *rate* specifies the proportion of partition data, which allocate to each client in the data island clinical setting. The number of proportions needs to be equal to the number of clients.
(10) *conditions* specifies the condition that select the data in the data island clinical setting, such as *conditions* = *file1*|*column1*> *value1*&*file1*|*column1*< *value2*, *file1*|*column1* = *value3*.
(11) *add* specifies the fine adjustment method of the data, which has been partitioned, such as *add* = *partition_number*@*file1*|*column1*> *conditions*.
(12) *image* specifies the column, which is image data in the file, such as *image* = *file*|*column*.
(12) *audio* specifies the column, which is audio data in the file, such as *audio* = *file*|*column*.
(12) *video* specifies the column, which is video data in the file, such as *video* = *file*|*column*.

4. The main API of Clinical AIBench

Clinical AIBench provides many python APIs for researchers to utilize the Clinical AIBench to train, test, and evaluate their AI models.

(1) *loadData(inipath, data_save_path, proportion = [80, 5, 15], setting = −1)*: The function of this method is loading the training set, validation set and test set from the local disk. Each parameter is defined as follows.
   (1) *inipath* specifies the path of the configuration file.
   (2) *data_save_path* specifies the saving path of data.
   (3) *proportion* specifies the partition proportion for the training set, validation set, and test set, and the default partition proportion is: 80% for the training set, 5% for the validation set, and 15% for the test set.
   (4) *setting* specifies the scenario type. It is the closed clinical setting or the data island clinical setting if the *setting* is −1. It is the open setting clinical if the *setting* is −2.

(2) *_toMatrix(source, save_path)*: The function of this method is obtaining feature matrix X and label matrix Y. Each parameter is defined as follows.
   (1) *source* specifies the data that has been downloaded.
   (2) *save_path* specifies the saving path that are the feature matrix X and the label matrix Y.

(3) *feature_extraction(input_path, image_tmp_save_path, image_save_path, pre_model = "DenseNet201", pooling = "avg", start = −1, end = −1)*: The function of this method is extracting features from a registered image by the pre-trained model. Each parameter is defined as follows.
   (1) *input_path* specifies the path of the image that needs extracting feature.
   (2) *image_tmp_save_path* specifies the temporary saving path of the image.
   (3) *image_save_path* specifies the saving path of the image, which was extracted feature.
   (4) *pre_model* specifies the pre-trained model.
   (5) *pooling* specifies the pooling method.
   (6) *start* and *end* specifies the identifier that respectively is the start and end position of the number of images.

(4) *convert_3Dto2DtoRGB(image_path, image_tmp_save_path, image_save_path)*: The function of this method is converting 3D image data to 2D image data, and then converting the gray image to RGB image. Each parameter is defined as follows.
   (1) *image_path* specifies the path of the image that needs to be converted.
   (2) *image_tmp_save_path* specifies the temporary saving path of the image.
   (3) *image_save_path* specifies the saving path of image, which was converted.

(5) *registration(input_path, out_path, image_type, weighted_image)*: The function of this method is aligning a image to a common template. Each parameter is defined as follows.
   (1) *image_path* specifies the path of the image that needs to be registered.
   (2) *output_path* specifies the saving path of the image, which was registered.
   (3) *image_type* specifies the type of the image.
   (4) *weighted_image* specifies the type of MRI-weighted image.

(6) *image_correct(image_path, image_save_path, image_type)*: The function of this method is correcting the non-uniform tissue intensity image into an image with uniform tissue intensity. Each parameter is defined as follows.
   (1) *image_path* specifies the path of the image that needs to be corrected.
   (2) *image_save_path* specifies the saving path of the image, which was corrected.
   (3) *image_type* specifies the type of the image.

(7) *strip_nonbrain(image_path, image_save_path, strip_method)*: The function of this method is stripping the non-brain structure components, such as the skull, neck, scalp and so on. Each parameter is defined as follows.
   (1) *image_path* specifies the path of the image that needs to be stripped.
   (2) *image_save_path* specifies the saving path of the image, which was stripped.
   (3) *strip_method* specifies the stripping method.

(8) *image_segmentation(image_path, image_save_path, segment_method)*: The function of this method is segmenting the image of preprocessed into different tissue types according to the different brain components. Each parameter is defined as follows.

   (1) *image_path* specifies the path of the image that needs to be segmented.
   (2) *image_save_path* specifies the saving path of the image, which was segmented.
   (3) *segment_method* specifies the segmentation method.

(9) *smoothing(image_path, image_save_path, smooth_method)*: The function of this method is compensating the inaccuracy of registration by reducing the image noise and making the image blur. Each parameter is defined as follows.

   (1) *image_path* specifies the path of the image that needs to be smoothed.
   (2) *image_save_path* specifies the saving path of the image, which was smoothed.
   (3) *smooth_method* specifies the smoothing method.

(10) *get_damage_level(method, predict_value, label)*: The function of this method is obtaining the damage value that the subject suffered during the medical diagnosis and treatment. These damage levels are scale from 0 to 5, and the 5 level is the highest. Each parameter is defined as follows.

   (1) *method* specifies the examination items of the subject during the diagnosis and treatment.
   (2) *predict_value* specifies the predicted value of the model. The predicted value is the disease type if it is a disease diagnosis prediction. The predicted value is the next treatment if it is a treatment prediction.
   (3) *label* specifies the actual disease type of the subject or the next treatment method advised by clinicians.

(11) *get_cost(method, predict_value, label)*: The function of this method is obtaining the cost of the subjects during the medical diagnosis and treatment. Each parameter is defined as follows.

   (1) *method* specifies the examination items of the subject during the diagnosis and treatment.
   (2) *predict_value* specifies the predicted value of the model. The predicted value is the disease type if it is a disease diagnosis prediction. The predicted value is the next treatment if it is a treatment prediction.
   (3) *label* specifies the actual disease type of the subject or the next treatment method advised by clinicians.

5. The Clinical AIBench use cases

In order to demonstrate the usage of Clinical AIBench, the diagnosis model of Alzheimer's disease is developed, verified, and evaluated in three different clinical settings, respectively.

5.1. Closed clinical setting

Currently, most clinical AI researches are based on the closed clinical setting: (1) The types of all subjects are already known, that is, the types of all subjects appear in the training set. (2) The data types of all subjects are the same. That is, all subjects contain all pre-specified types of data. (3) All medical institutions can obtain all pre-specified types of data, that is, all medical institutions have pre-specified capabilities of examination and treatment.

In this paper, according to the characteristics of the restricted scenario, we construct a closed clinical setting based on the basic information of Alzheimer's disease subjects and MRI images: (1) The clinical setting contains 2127 subjects, of which 740 are Alzheimer's disease (AD), 1082 are mild cognitive impairment (MCI), and 589 are cognitively normal subjects (CN). (2) All types of subjects are divided into training set, verification set, and test set at a ratio of 80%:5%:15%. (3) Because subjects generally have multiple follow-up visits, Clinical AIBench stipulates that the data of different visits of the same subject is only allowed to appear in one of the training set, the verification set, and the test set. (4) All subjects contain basic information and MRI information. When a subject is missing some part of the data, Clinical AIBench will fill in the data according to a specified strategy.

To construct the clinical setting above, we create a configuration file as shown in Fig. 4.

In this paper, we develop and evaluate a diagnosis model of Alzheimer's disease in above clinical setting. As shown in Fig. 5, it is a model of Alzheimer's classification based on Keras. As shown in Fig. 6, it is the entry point of the program. Researchers can start the *main()* function to execute the entire program. Line 21 of Fig. 6 is to create a model object. Line 34 of Fig. 6 is the model evaluation. Line 39 of Fig. 6 is the model prediction.

5.2. Data island clinical setting

Due to data privacy, data security, and data value, clinical data is difficult to share for researchers. The clinical data is usually stored in their medical institution, forming the data island clinical setting. In order to promote the progress of AI technologies in the data island clinical setting, new machine learning concepts (federated learning and swarm learning) are proposed [12–14].

In this paper, we construct a data island clinical setting: (1) The subject and data types are similar to the closed clinical setting. (2) Divide the clinical data into multiple parts according to different medical institutions to form a natural data island clinical setting.

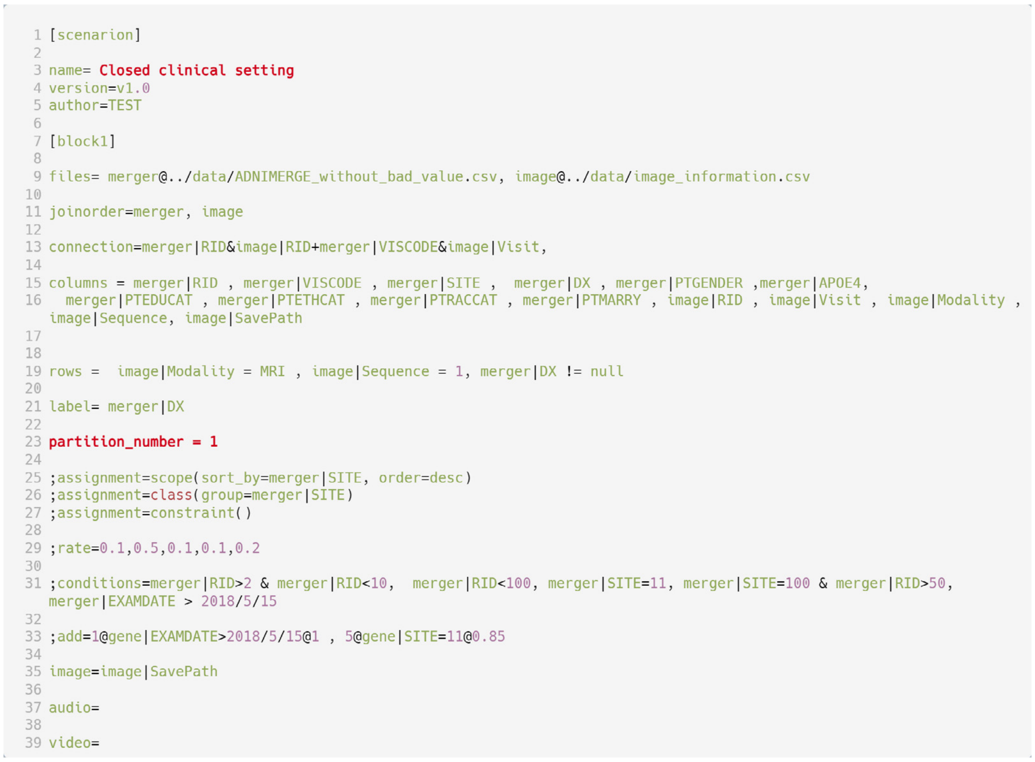
To construct the clinical setting above, we create a configuration file as shown in Fig. 7. The difference between this scenario configuration file and the above closed scenario configuration file is that the number of clients is sets to −1 (in line 23 of Fig. 7), and the data partition method selected the *_class* (in line 26 of Fig. 7).

In this paper, we develop and evaluate a diagnosis model of Alzheimer's disease in above clinical setting. As shown in Fig. 8, it was a model of Alzheimer's classification based on Pytorch. As shown in Fig. 9, it is the entry point of the program. The 46 line of Fig. 9 is the entrance to the model training [11].

5.3. Open clinical setting

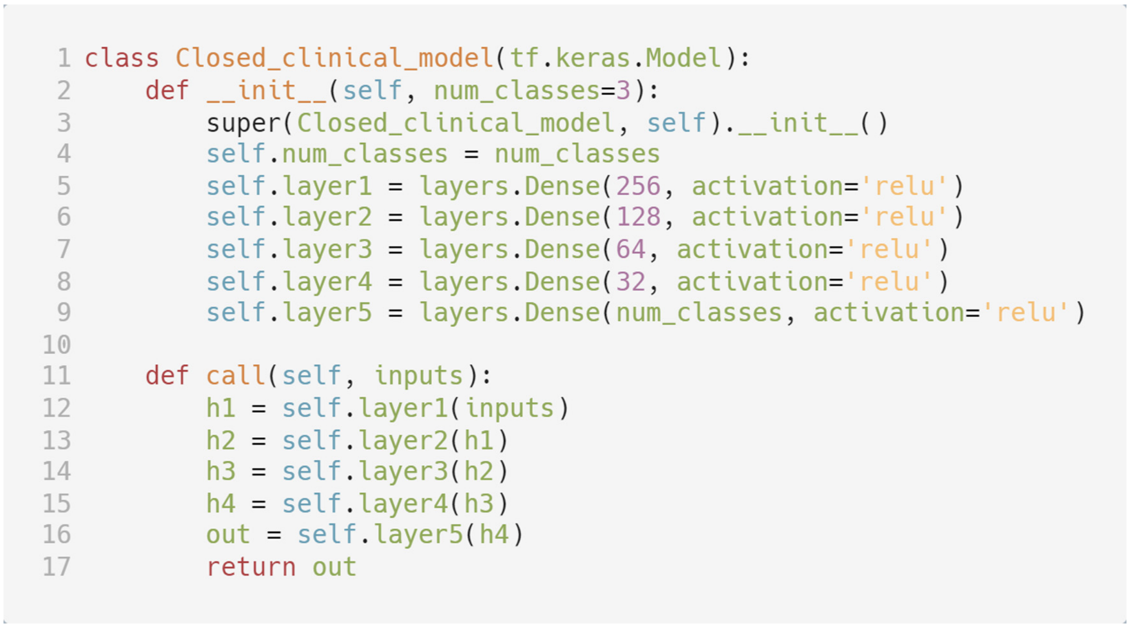
As mentioned above, the real-world clinical setting is open and full of complexity and uncertainty. We capture the characteristics of the real-world clinical setting and construct the open clinical setting which maintains the main features of the real-world clinical setting: (1) The type of some subjects is unknown, that is, some type of subject in the test set does not appear in the training set and verification set. (2) The specific conditions of the subjects are different, and the examinations and treatment methods executed on the subjects are different, that is, the data types of the subjects are different. (3) The medical conditions of medical institutions are different. Some institutions can meet the requirements of the diagnosis and treatment strategies of subjects, while others cannot. That is to say, the types of data of subjects collected in different medical institutions are many different.

In this paper, we do not provide the modifiable function of the open clinical setting. And we develop and evaluate a diagnosis model of Alzheimer's disease in the above clinical setting. Clinical AIBench proposes a unified data representation framework for the open clinical setting of Alzheimer's disease, since the different dimensions of each data category, the number of data categories included in each visit are different, and the number of history visits included in each subject is also different. The data framework presents an examination category



```
[scenarion]

name= Closed clinical setting
version=v1.0
author=TEST

[block1]

files= merger@../data/ADNIMERGE_without_bad_value.csv, image@../data/image_information.csv

joinorder=merger, image

connection=merger|RID&image|RID+merger|VISCODE&image|Visit,

columns = merger|RID , merger|VISCODE , merger|SITE ,  merger|DX , merger|PTGENDER ,merger|APOE4,
  merger|PTEDUCAT , merger|PTETHCAT , merger|PTRACCAT , merger|PTMARRY , image|RID , image|Visit , image|Modality , image|Sequence, image|SavePath


rows =  image|Modality = MRI , image|Sequence = 1, merger|DX != null

label= merger|DX

partition_number = 1

;assignment=scope(sort_by=merger|SITE, order=desc)
;assignment=class(group=merger|SITE)
;assignment=constraint()

;rate=0.1,0.5,0.1,0.1,0.2

;conditions=merger|RID>2 & merger|RID<10,  merger|RID<100, merger|SITE=11, merger|SITE=100 & merger|RID>50, merger|EXAMDATE > 2018/5/15

;add=1@gene|EXAMDATE>2018/5/15@1 , 5@gene|SITE=11@0.85

image=image|SavePath

audio=

video=
```

Fig. 4. Closed clinical setting configuration file.



```python
class Closed_clinical_model(tf.keras.Model):
    def __init__(self, num_classes=3):
        super(Closed_clinical_model, self).__init__()
        self.num_classes = num_classes
        self.layer1 = layers.Dense(256, activation='relu')
        self.layer2 = layers.Dense(128, activation='relu')
        self.layer3 = layers.Dense(64, activation='relu')
        self.layer4 = layers.Dense(32, activation='relu')
        self.layer5 = layers.Dense(num_classes, activation='relu')

    def call(self, inputs):
        h1 = self.layer1(inputs)
        h2 = self.layer2(h1)
        h3 = self.layer3(h2)
        h4 = self.layer4(h3)
        out = self.layer5(h4)
        return out
```

Fig. 5. Closed clinical setting model.

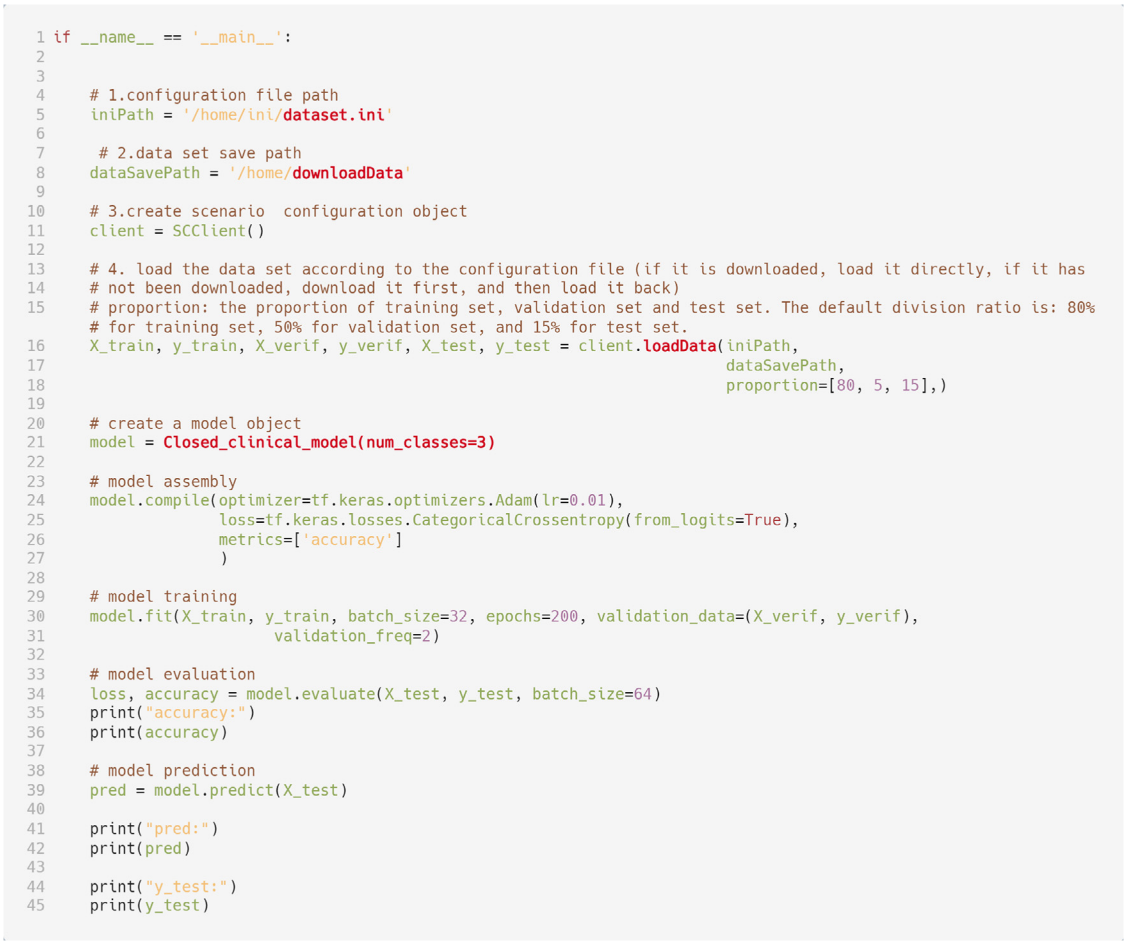
in the subject's visit by an array with a shape of 1×2090. The shape of our data is $n \times 2090$, n is the number of categories of data for the subject. The data of the open clinical setting of Alzheimer's disease is able downloaded by Clinical AIBench. And as a sample, we develop a model based on the open clinical setting of Alzheimer's disease shown in the file OpenClinicalAI.py and HierarchicalOpenNet.py [15].

6. The clinical data in the Clinical AIBench

Clinical AIBench is an open and evolving benchmark. Datasets that we select will follow the following principles (1): diseases that are more concerned in the current clinical research field; (2): datasets that are more complete; (3): and datasets that are low acquisition cost. Currently, it contains three clinical datasets: Alzheimer's disease (more complete), COVID-19 (more concerned), and dental (low acquisition cost). In addition, the ICU and psychiatric disorders datasets will be added to Clinical AIBench soon.

6.1. Privacy and data security

The clinical data is very sensitive, involving subject privacy and data security. According to the security objectives in Section 5 of the



```python
if __name__ == '__main__':


    # 1.configuration file path
    iniPath = '/home/ini/dataset.ini'

     # 2.data set save path
    dataSavePath = '/home/downloadData'

    # 3.create scenario  configuration object
    client = SCClient()

    # 4. load the data set according to the configuration file (if it is downloaded, load it directly, if it has
    # not been downloaded, download it first, and then load it back)
    # proportion: the proportion of training set, validation set and test set. The default division ratio is: 80%
    # for training set, 50% for validation set, and 15% for test set.
    X_train, y_train, X_verif, y_verif, X_test, y_test = client.loadData(iniPath,
                                                                         dataSavePath,
                                                                         proportion=[80, 5, 15],)

    # create a model object
    model = Closed_clinical_model(num_classes=3)

    # model assembly
    model.compile(optimizer=tf.keras.optimizers.Adam(lr=0.01),
                  loss=tf.keras.losses.CategoricalCrossentropy(from_logits=True),
                  metrics=['accuracy']
                  )

    # model training
    model.fit(X_train, y_train, batch_size=32, epochs=200, validation_data=(X_verif, y_verif),
                        validation_freq=2)

    # model evaluation
    loss, accuracy = model.evaluate(X_test, y_test, batch_size=64)
    print("accuracy:")
    print(accuracy)

    # model prediction
    pred = model.predict(X_test)

    print("pred:")
    print(pred)

    print("y_test:")
    print(y_test)
```

Fig. 6. Closed clinical setting's main function.

standard GB/T 39725-2020 "Information security technology-Guide for health data security" [16], researchers need to ensure the confidentiality, integrity, and availability of clinical data and ensure that the usage of the clinical data is legal. In addition, in accordance with the standard GB/T 39725-2020 "Information security technology-Guide for health data security" Section 7 as mentioned in point 4 of point b, when restricted datasets are used for scientific research, medical/health education, and public health purposes, the corresponding personal health, and medical data can be used without the authorization of the subject. Thus the public data of Clinical AIBench currently only provides services for scientific research.

The private data of Clinical AIBench will be strictly desensitized according to the Health Insurance Portability and Accountability Act/1996, Public Law 104–191 and technical standards. In addition, all private data is stored in the private server of the data owner, and provides services through Clinical AIBench with the data island clinical setting manner.

6.2. Alzheimer's disease

The data support of Alzheimer's disease clinical setting is derived from the ADNI dataset, as shown in Table 1, which contains 9591 individual visits of 2127 subjects [17]. We organize the ADNI dataset according to the subject identification (RID) and the visit code (VISCODE2, which represents the time between the current visit date of subjects and the first visit date). The dataset is divided into the following parts.

(1) The merge table has 113 fields, including the classification of subjects and the information currently generally considered to be related to Alzheimer's disease, and the value corresponding to the first visit of that information. Full merge table information is available at the website Merge table, where for unified expression and understanding field VISCODE is expressed as VISCODE2.
(2) The basic information table has 148 fields, including the subject's demographic information, family medical history, personal medical history, and the subject's current symptoms. More information is available at Basic information table.
(3) The physical examination information table contains 39 fields, including the subject's neurological examination information, physical examination information, and vital signs information. More information is available at Physical examination information table.
(4) The cognitive information table has 259 fields, including Alzheimer's Disease Assessment Scale-Cognitive information (ADASCog), Mini Mental State Exam information (MMSE), Montreal Cognitive Assessment information (MoCA), Clinical Dementia Rating information (CDR), and Cognitive Change Index information (CCI). More information is available at Cognitive information table.
(5) The cognitive test information table has 73 fields, including neuropsychological information: Clock Painting Test, Logic Memory Test-I (Instant Memory), Logic Memory Test-II (Time-lapse Memory), Rey Hearing Test, Wired Test, Animal Category Test, Boston Naming Test, American Adult Reading Test. More information is available at Cognitive test information table.
(6) The function and behavior test table has 112 fields, including Functional Assessment Questionnaire information (FAQ), Everyday Cognition — Participant Self-Report information

```
[scenarion]

name= Data island clinical setting
version=v1.0
author=TEST

[block1]

files= merger@../data/ADNIMERGE_without_bad_value.csv, image@../data/image_information.csv

joinorder=merger, image

connection=merger|RID&image|RID+merger|VISCODE&image|Visit,

columns = merger|RID , merger|VISCODE , merger|SITE ,  merger|DX , merger|PTGENDER ,merger|APOE4,
  merger|PTEDUCAT , merger|PTETHCAT , merger|PTRACCAT , merger|PTMARRY , image|RID , image|Visit , image|Modality , image|Sequence, image|SavePath


rows =  image|Modality = MRI , image|Sequence = 1, merger|DX != null

label= merger|DX

partition_number = -1

;assignment=scope(sort_by=merger|SITE, order=desc)
assignment=class(group=merger|SITE)
;assignment=constraint()

rate=0.1,0.5,0.1,0.1,0.2

;conditions=merger|RID>2 & merger|RID<10,  merger|RID<100, merger|SITE=11, merger|SITE=100 & merger|RID>50, merger|EXAMDATE > 2018/5/15

;add=1@gene|EXAMDATE>2018/5/15@1 , 5@gene|SITE=11@0.85

image=image|SavePath

audio=

video=
```

Fig. 7. Data island clinical setting configuration file.

```python
class Data_island_clinical_model(nn.Module):
    def __init__(self, num_classes=3):
        super(Data_island_clinical_model, self).__init__()
        self.fc1 = nn.Linear(1930, 200)
        self.fc2 = nn.Linear(200, 200)
        self.fc3 = nn.Linear(200, num_classes)
        self.relu = nn.ReLU()

    def forward(self, x):
        x = x.view(x.shape[0], -1)
        x = self.relu(self.fc1(x))
        x = self.relu(self.fc2(x))
        x = self.fc3(x)
        return x
```

Fig. 8. Data island clinical setting model.

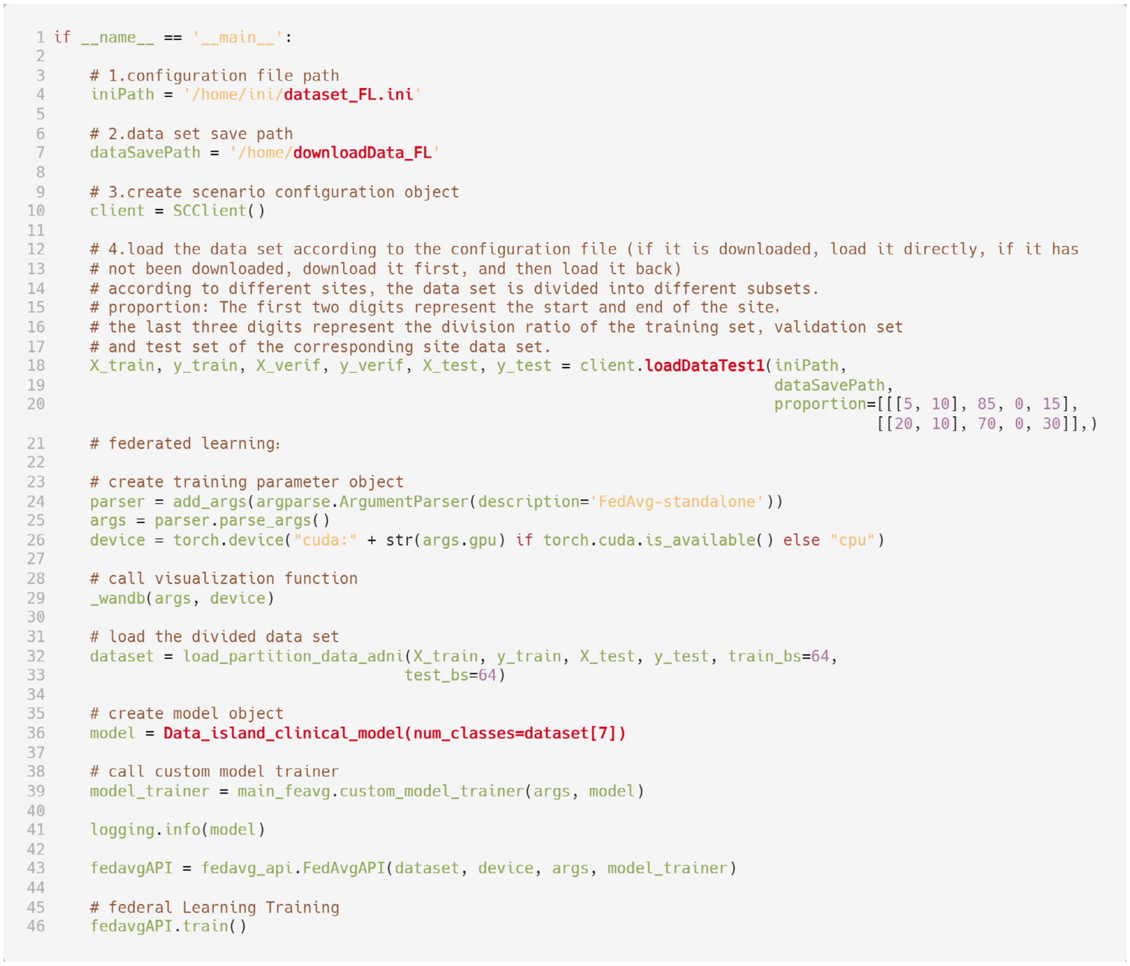
(ECOGPT), and Everyday Cognition — Study Partner Report information (ECOGSP). More information is available at Function and behavior table.

(7) The psychiatric test table has 55 fields, including the subject's Geriatric Depression Scale information (GDS) and Neuropsychiatric Inventory Examination information (NPI). More information is available at Psychiatric test table.

(8) The blood table has 459 fields, including the subject's various blood test information. More information is available at Blood table.

(9) The urine table has 4 fields, including the subject's various urine test information. More information is available at Urine table.

(10) The cerebral spinal fluid (CSF) table has 379 fields, including the subject's various cerebrospinal fluid test informations. More information is available at Cerebral spinal fluid table.



```
if __name__ == '__main__':

    # 1.configuration file path
    iniPath = '/home/ini/dataset_FL.ini'

    # 2.data set save path
    dataSavePath = '/home/downloadData_FL'

    # 3.create scenario configuration object
    client = SCClient()

    # 4.load the data set according to the configuration file (if it is downloaded, load it directly, if it has
    # not been downloaded, download it first, and then load it back)
    # according to different sites, the data set is divided into different subsets.
    # proportion: The first two digits represent the start and end of the site.
    # the last three digits represent the division ratio of the training set, validation set
    # and test set of the corresponding site data set.
    X_train, y_train, X_verif, y_verif, X_test, y_test = client.loadDataTest1(iniPath,
                                                                          dataSavePath,
                                                                          proportion=[[[5, 10], 85, 0, 15],
                                                                                      [[20, 10], 70, 0, 30]],)
    # federated learning:

    # create training parameter object
    parser = add_args(argparse.ArgumentParser(description='FedAvg-standalone'))
    args = parser.parse_args()
    device = torch.device("cuda:" + str(args.gpu) if torch.cuda.is_available() else "cpu")

    # call visualization function
    _wandb(args, device)

    # load the divided data set
    dataset = load_partition_data_adni(X_train, y_train, X_test, y_test, train_bs=64,
                                      test_bs=64)

    # create model object
    model = Data_island_clinical_model(num_classes=dataset[7])

    # call custom model trainer
    model_trainer = main_feavg.custom_model_trainer(args, model)

    logging.info(model)

    fedavgAPI = fedavg_api.FedAvgAPI(dataset, device, args, model_trainer)

    # federal Learning Training
    fedavgAPI.train()
```

Fig. 9. Data island clinical setting's main function.

(11) The gene table has 351 fields, including subject's single nucleotide polymorphisms (SNPs) related to Alzheimer's disease. Only one genetic test was performed per subject in Clinical AIBench, so the genetic test table does not contain visit codes and defaults to the data at the first visit. In addition, each SNP is represented by 5 fields, which are the four bases A, G, C, T, and the SNP's confidence. More information is available at Gene table.

(12) The medical imaging table has 33 fields, including the subject's 3 types of original medical images and information: MRI, PET-18FDG, PET-AV45. More information is available at Medical imaging table.

6.3. COVID-19

The COVID-19 data support in Clinical AIBench is collected from different publicly accessible datasets, online resources, and published papers. The COVID-19 dataset contains three different publicly available datasets: covid-chestxray-dataset [18], Figure1-COVID-chestxray-dataset [19], and Actualmed-COVID-chestxray-dataset [20].

As shown in Table 2, which contains 1243 individual visit records of 742 subjects. The summary table of COVID-19 has 24 fields, including the demographics of subject, current physical condition of subject, original medical image, and image type. More information is available at COVID-19 table.

6.4. Dental

The dental dataset is a private dataset from cooperative medical institutions that was desensitized. The dental dataset can only be used to develop, validate, and evaluate clinical AI models or systems similar to the data island clinical setting, and runs only on the private server of the data owner.

As shown in Table 3, the dental dataset contains 661 individual visits of 447 subjects. The dental data table has 5 fields, including the id, data of visit, age, gender, saving path of image.

7. Precautions for using Clinical AIBench

Clinical AIBench can be accessed according to the standard data usage agreement (different clinical settings involve different usage agreements). According to this agreement, users must agree to only use Clinical AIBench for the purposes stated in the agreement. The users of Clinical AIBench should thank the original authors and research laboratories for their contributions by correctly citing related article and their links to Clinical AIBench.

To objectively, fairly, and impartially evaluate the clinical AI models and systems of researchers. Clinical AIBench as an independent third party tool, which will provide a consistent clinical setting for evaluation. The configuration file of Clinical AIBench provided clinical setting is not allowed to be modified.

The data of Clinical AIBench comes from two sources: public datasets and private datasets. The public datasets can be downloaded for the development, verification, and evaluation of clinical models or systems of researchers. However, the private datasets can only be used to develop, validate, and evaluate clinical AI models or systems similar to the data island clinical setting; that is, datasets are stored on the private server of the data owner and cannot be downloaded.

Table 1
Characteristics of Alzheimer's disease subjects.

		Subjects
Age	54–59.9	80
	60–69.9	596
	70–70.9	1048
	80–80.9	395
	90–91.9	6
	Unknow	2
Gender	Female	1130
	Male	997
Educate	4–7	11
	8–10	40
	11–13	353
	14–16	823
	17–20	900
Ethnic category	Hisp/Latino	73
	Not Hisp/Latino	2042
	Unknown	12
Racial category	Asian	40
	Black	88
	Hawaiian/Other PI	2
	More than one	25
	White	1964
	Am Indian/Alaskan	4
	Unknown	4
Marriage	Married	1618
	Never_married	73
	Widowed	238
	Divorced	191
	Unknown	7
Category	AD	740
	CN	589
	MCI	1082
	SMC	280

Table 2
Summary of COVID-19 subjects.

		Subjects
Age	18–30	48
	31–50	122
	51–70	134
	71–94	82
	Unknown	356
Sex	Female	169
	Male	281
	Unknown	292
Offset	0–20	312
	21–40	10
	41–60	4
	<0	10
	61–365	3
	Unknown	403
Category	No finding	132
	Pneumonia	133
	COVID-19	380
	Unknown	97

Table 3
Characteristic of dental subjects.

		Subjects
Sex	Female	242
	Male	205
Age	0–20	97
	21–40	209
	41–60	90
	61–80	49
	81–90	2

8. Conclusion

Many clinical AI algorithms have been proposed and achieved excellent performances in the clinical fields in the closed clinical and data island settings. In contrast, few clinical AI algorithms are applied in the real-world clinical setting. Clinical AIBench, as a benchmark suite or third-party tool, provides open and fair clinical settings for evaluating clinical AI algorithms. In addition, Clinical AIBench also provides a configurable clinical environment for developing clinical AI algorithms.

Clinical AIBench is an open and continuously evolving benchmark suite. Clinical AIBench will add the ICU and psychiatric disorders datasets soon.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is supported by the Project of Guangxi Science and Technology, China (No. GuiKeAD20297004 to Y.H.), and the National Natural Science Foundation of China (No. 61967002 to S.T.).

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, Imagenet: A large-scale hierarchical image database, in: 2009 IEEE Conference on Computer Vision and Pattern Recognition, Ieee, 2009, pp. 248–255.
[2] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.
[3] S.G. Mueller, M.W. Weiner, L.J. Thal, R.C. Petersen, C.R. Jack, W. Jagust, J.Q. Trojanowski, A.W. Toga, L. Beckett, Ways toward an early diagnosis in alzheimer's disease: the alzheimer's disease neuroimaging initiative (adni), Alzheimer's Dement. 1 (1) (2005) 55–66.
[4] K. Tomczak, P. Czerwińska, M. Wiznerowicz, The cancer genome atlas (tcga): an immeasurable source of knowledge, Contemp. Oncol. 19 (1A) (2015) A68.
[5] S. Benjamens, P. Dhunnoo, B. Meskó, The state of artificial intelligence-based fda-approved medical devices and algorithms: an online database, NPJ Digit. Med. 3 (1) (2020) 1–8.
[6] P. Brocklehurst, D. Field, K. Greene, E. Juszczak, R. Keith, S. Kenyon, L. Linsell, C. Mabey, M. Newburn, R. Plachcinski, et al., Computerised interpretation of fetal heart rate during labour (infant): a randomised controlled trial, Lancet 389 (10080) (2017) 1719–1729.
[7] Y. Liang, Y. Guo, Y. Gong, C. Luo, J. Zhan, Y. Huang, Flbench: A benchmark suite for federated learning, in: BenchCouncil International Federated Intelligent Computing and Block Chain Conferences, Springer, 2020, pp. 166–176.
[8] W. Gao, F. Tang, J. Zhan, X. Wen, L. Wang, Z. Cao, C. Lan, C. Luo, X. Liu, Z. Jiang, Aibench scenario: Scenario-distilling ai benchmarking, in: 2021 30th International Conference on Parallel Architectures and Compilation Techniques (PACT), IEEE, 2021, pp. 142–158.
[9] J. Zhan, Call for establishing benchmark science and engineering, BenchCouncil Trans. Benchmarks Stand. Eval. 1 (1) 100012.
[10] F. Zhang, C. Luo, C. Lan, J. Zhan, Benchmarking feature selection methods with different prediction models on large-scale healthcare event data, BenchCouncil Trans. Benchmarks Stand. Eval. 1 (1) (2021) 100004.
[11] C. He, S. Li, J. So, M. Zhang, H. Wang, X. Wang, P. Vepakomma, A. Singh, H. Qiu, L. Shen, P. Zhao, Y. Kang, Y. Liu, R. Raskar, Q. Yang, M. Annavaram, S. Avestimehr, Fedml: A research library and benchmark for federated machine learning, arXiv preprint arXiv:2007.13518.
[12] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, Y. Khazaeni, Bayesian nonparametric federated learning of neural networks, in: International Conference on Machine Learning, PMLR, 2019, pp. 7252–7261.
[13] C. Xie, K. Huang, P.-Y. Chen, B. Li, Dba: Distributed backdoor attacks against federated learning, in: International Conference on Learning Representations, 2019.
[14] S. Warnat-Herresthal, H. Schultze, K.L. Shastry, S. Manamohan, S. Mukherjee, V. Garg, R. Sarveswara, K. Händler, P. Pickkers, N.A. Aziz, et al., Swarm learning for decentralized and confidential clinical machine learning, Nature 594 (7862) (2021) 265–270.

[15] Y. Huang, N. Wang, S. Tang, L. Ma, T. Hao, Z. Jiang, F. Zhang, G. Kang, X. Miao, X. Guan, et al., Openclinicalai: enabling ai to diagnose diseases in real-world clinical settings, arXiv preprint arXiv:2109.04004.

[16] Inspection, G.A.o.Q.S., Quarantine of the People's Republic of China, Standardization administration, http://std.samr.gov.cn//gb/search/gbdetailed?id=b691bb77876cd126e05397be0a0af3b3.

[17] S.G. Mueller, M.W. Weiner, L.J. Thal, R.C. Petersen, C. Jack, W. Jagust, J.Q. Trojanowski, A.W. Toga, L. Beckett, The alzheimer's disease neuroimaging initiative, Neuroimaging Clin. N. Am. 15 (4) (2005) 869.

[18] J.P. Cohen, P. Morrison, L. Dao, Covid-19 image data collection, arXiv:2003.11597. URL https://github.com/ieee8023/covid-chestxray-dataset.

[19] agchung, Figure1 Covid-19 Chest X-Ray Dataset Initiative, GitHub repository, URL https://github.com/agchung/Figure1-COVID-chestxray-dataset.

[20] p. agchung, lindawangg Linda Wang, Actualmed Covid-19 Chest X-Ray Dataset, GitHub repository, URL https://github.com/agchung/Actualmed-COVID-chestxray-dataset.

TBench Editorial Board

More information about this series at https://www.benchcouncil.org/tbench/

TBench Call For Papers

BenchCouncil Transactions on Benchmarks, Standards and Evaluations (TBench)
ISSN:2772-4859

Aims and Scopes

BenchCouncil Transactions on Benchmarks, Standards, and Evaluations (TBench) publishes position articles that open new research areas, research articles that address new problems, methodologies, tools, survey articles that build up comprehensive knowledge, and comments articles that argue the published articles. The submissions should deal with the benchmarks, standards, and evaluation research areas. Particular areas of interest include, but are not limited to:

- 1. Generalized benchmark science and engineering (see https://www.sciencedirect.com/science/article/pii/S2772485921000120), including but not limited to
  - measurement standards
  - standardized data sets with defined properties
  - representative workloads
  - representative data sets
  - best practices
- 2. Benchmark and standard specifications, implementations, and validations of:
  - Big Data
  - AI
  - HPC
  - Machine learning
  - Big scientific data
  - Datacenter
  - Cloud
  - Warehouse-scale computing
  - Mobile robotics
  - Edge and fog computing
  - IoT
  - Chain block
  - Data management and storage
  - Financial domains
  - Education domains
  - Medical domains
  - Other application domains
- 3. Data sets
  - Detailed descriptions of research or industry datasets, including the methods used to collect the data and technical analyses supporting the quality of the measurements.
  - Analyses or meta-analyses of existing data and original articles on systems, technologies, and techniques that advance data sharing and reuse to support reproducible research.
  - Evaluating the rigor and quality of the experiments used to generate the data and the completeness of the data description.
  - Tools generating large-scale data while preserving their original characteristics.
- 4. Workload characterization, quantitative measurement, design, and evaluation studies of:
  - Computer and communication networks, protocols, and algorithms
  - Wireless, mobile, ad-hoc and sensor networks, IoT applications
  - Computer architectures, hardware accelerators, multi-core processors, memory systems, and storage networks
  - High-Performance Computing
  - Operating systems, file systems, and databases

  - Virtualization, data centers, distributed and cloud computing, fog, and edge computing
  - Mobile and personal computing systems
  - Energy-efficient computing systems
  - Real-time and fault-tolerant systems
  - Security and privacy of computing and networked systems
  - Software systems and services, and enterprise applications
  - Social networks, multimedia systems, Web services
  - Cyber-physical systems, including the smart grid
- 5. Methodologies, metrics, abstractions, algorithms, and tools for:
  - Analytical modeling techniques and model validation
  - Workload characterization and benchmarking
  - Performance, scalability, power, and reliability analysis
  - Sustainability analysis and power management
  - System measurement, performance monitoring, and forecasting
  - Anomaly detection, problem diagnosis, and troubleshooting
  - Capacity planning, resource allocation, run time management, and scheduling
  - Experimental design, statistical analysis, simulation
- 6. Measurement and evaluation
  - Evaluation methodology and metric
  - Testbed methodologies and systems
  - Instrumentation, sampling, tracing, and profiling of Large-scale real-world applications and systems
  - Collection and analysis of measurement data that yield new insights
  - Measurement-based modeling (e.g., workloads, scaling behavior, assessment of performance bottlenecks)
  - Methods and tools to monitor and visualize measurement and evaluation data
  - Systems and algorithms that build on measurement-based findings
  - Advances in data collection, analysis, and storage (e.g., anonymization, querying, sharing)
  - Reappraisal of previous empirical measurements and measurement-based conclusions
  - Descriptions of challenges and future directions the measurement and evaluation community should pursue