

Research Article

English Language Features in Linguistics by High-Performance Computing

Dongyan Chen ^{1,2}, **Suryani Awang**³ and **Zaemah Abdul Kadir**²

¹College of International Studies, Beibu Gulf University, Qinzhou 535015, Guangxi, China

²Akademi Pengajian Bahasa Shah Alam, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia

³Akademi Pengajian Bahasa, Universiti Teknologi MARA Cawangan Kelantan, Bukit Ilmu 18500, Machang, Kelantan, Malaysia

Correspondence should be addressed to Dongyan Chen; chendonngyan@bbgu.edu.cn

Received 11 April 2022; Revised 18 May 2022; Accepted 1 June 2022; Published 16 June 2022

Academic Editor: Yanyi Rao

Copyright © 2022 Dongyan Chen et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

High-performance computing clusters are mainly used to deal with complex computing problems and are widely used in the fields of meteorology, ocean, environment, life science, and computer-aided engineering. Language is the way humans communicate and communicate. Linguistic features are the stylistic features that distinguish all languages from other languages. This paper aims to study how to analyze English language features based on high-performance computing. This paper addresses the problem of linguistic feature analysis, which is built on high-performance computing. Therefore, this paper expounds the related concepts and algorithms, and designs and analyzes the characteristics of English language. The experimental results show that among the 160 English sentences in two different journals, complex sentences are the most used, with a total of 55 sentences, accounting for 34.38%. The second is mixed sentence types, 47 of which are mixed sentence structures, accounting for 29.38%. Among them, the combination of simple sentences + coordinating complex sentences + complex sentences constitutes the most mixed sentences, which appear 12 times and 8 times in *ELT Journal* and *SSCI*, respectively, accounting for 15.00% and 10.00% of their respective corpora.

1. Introduction

In the face of high concurrency, multicomputing models, and high-performance computing under big data storage, the timeliness of data and user response cannot be truly improved. How to allocate tasks and call resources for high-performance clusters is the key to improving performance. Among the more than 5,000 languages in the world, English is the most widely spoken language. For half a century, English has become the lingua franca. It is estimated that by the end of the twenty-first century, the total number of English speakers in the world will reach 2.1 billion. From the Eastern Hemisphere to the Western Hemisphere, one can hear all kinds of “English,” which means that there are many varieties of English. English in the twenty-first century has become a multiethnic, multicultural, and multifunctional international language. Besides native speakers, many non-native speakers also use it for international and domestic

communication. The globalization of English has led to its extensive localization, and the concepts of English Variations and World Englishes have emerged around the world.

Language is the carrier of human thinking, a necessary tool for people to communicate with each other, and an integral part of human civilization. Linguistic features have been widely used in article style analysis, and quantitative analysis of language features can be used to distinguish the style and type of text, and to analyze the correlation between language features and quality of text. The national language features are the symbols and symbols of the national spirit. The research and analysis of English language features provide a broad space for the emergence and development of English.

The innovations of this paper are as follows: (1) This paper combines language features with high-performance algorithms and introduces the theory and related methods of high-performance algorithms in detail. This paper mainly

introduces the parallel algorithm and GPU-based parallel ant colony optimization algorithm. (2) In the face of analyzing language features, this paper classifies the structure of sentences. This paper compares the language features in different journals and concludes that the mixed sentence type is the mainstream of English language features.

2. Related Work

With the progress of society, more and more people have studied high-performance computing. Interactive high-performance computing undoubtedly benefits many computational science and engineering applications when simulation results need to be visualized in real time, that is, during computation. However, interactive HPC presents a number of new challenges that must be addressed - one of which is solving the problem of fast and efficient data transfer between the simulation backend and the visualization frontend. Because gigabytes of data per second is not uncommon for simulations running on around (hundred thousand) cores. Mundani et al. [1] introduced a new method based on sliding windows and small-scale simulations, which can address any limitations of the user on interactive windows [1]. The problem of simulating microscale urban traffic in large-scale environments presents a great opportunity for the utilization of HPC systems. Parallel implementation of such computations (which must synchronize complex data-intensive processing) is not trivial. The simulation proposed by Turek W is based on the concept of controlled desynchronization of computation, which does not violate the model. The implementation in Erlang language uses the Erlang distribution mechanism to build and manage computing clusters [2]. The complexity and uncertainty of bridge construction projects require simulation analysis and planning for these projects. On the other hand, optimization can be used to address the inverse relationship between project cost and time, and find the right trade-off between these two key factors. Furthermore, the large amount of resources required for large bridge construction projects results in a very large search space. Therefore, it is necessary to use parallel computing to reduce the computation time of simulation-based optimization problems. Another problem in this area is that most construction simulation tools require an integrated platform to combine with optimization techniques. To alleviate these limitations, Salimi et al. [3] develop a simulation-based integrated optimization framework on a high-performance computing (HPC) platform and analyze its performance through case studies. They employ a master-slave (or global) parallel genetic algorithm (GA) to reduce computation time and efficiently utilize the full capacity of the computer. In addition, sensitivity analysis is applied to determine promising genetic algorithm configurations and the optimal number of cores for parallel use, and to analyze the impact of genetic algorithm parameters on the overall performance of the simulation-based optimization model [3]. The Neuroscience Initiative aims to develop new techniques and tools to measure and manipulate neuronal circuits. To handle the large volumes of data generated by these tools, Bouchard et al. [4] envisions co-locating open data

repositories in standard formats with high-performance computing hardware using open-source optimized analysis code [4]. Hybrid cloud has gained popularity in various organizations in recent years due to its ability to provide additional capacity in public cloud, augmenting private cloud capacity when needed. However, scheduling jobs for distributed applications on hybrid cloud resources brings new challenges. A key concern is the risk of exposing private data and jobs in third-party public cloud infrastructure. The problem to be solved by Sharif et al. [5] is to design workflow scheduling algorithms to meet client deadlines without compromising data and task privacy requirements. The work of Sharif et al. [5] differs from most studies on workflow scheduling, where the main goal of workflow scheduling is to achieve a balance between ideal but incompatible constraints, such as meeting deadlines and/or minimizing execution time [5]. Noda et al. [6] will present the roadmap and research questions related to multiagent social simulation to illustrate the direction of technological achievements in this field [6]. However, the shortcoming of these studies is that the problems arising from high-performance computing are not properly dealt with.

3. Approaches to High-Performance Computing

3.1. Theoretical Basis of HPC. High-performance computing (HPC) refers to a computer system and environment consisting of multiple processors or multiple computers in a high-performance cluster. It can provide much higher computing power than traditional computers for large-scale data analysis and processing supercomputers [7].

With the continuous development of application requirements, the improvement of existing computer speed cannot keep up with the increase in computing speed requirements, especially in complex scientific computing, digital model analysis, simulation, engineering problems, and other application fields that require large-scale data volume and complex computing. All calculations and processing are required to be completed within an acceptable time [8]. However, it is limited to further increase the operation speed of a single processor. Therefore, the research on high-performance computing technology focuses on the development of supercomputers, and the study of parallel computing algorithms and software. Both high-performance computing and cloud computing models have their strengths and weaknesses. Table 1 summarizes some key features of HPC and cloud computing, and it can be seen that no single model can be the best solution for all features.

In the traditional high-performance computing model, computing workloads are processed in a well-managed and secure environment. However, computing capacity is fixed and rarely supports virtualization and resource sharing.

In view of the respective advantages and disadvantages of high-performance computing and cloud computing, many systems seek to improve these two computing modes, most of which are combined with grid computing. Combining high-performance computing and grid computing models is a practice in many scientific workflows to increase the

TABLE 1: Comparison of key features of HPC and cloud computing.

Characteristic	High performance computing	Cloud computing
Calculation quantity	Changeless	Increase in demand elasticity
Computational performance	Very high	Below average
Support virtual machine	Little support	Always support
Resource heterogeneity	Very low	Below average
Interoperability	Not applicable	Lower
Security	High	Below average

computational volume of high-performance computing by combining distributed grid resources. This method has been widely used in many projects [9, 10]. For computing tasks with computing performance as the bottleneck, this paper proposes a scientific cloud such as grid cloud. It is oriented towards scientific computing needs and implements infrastructure-as-a-service cloud computing solutions through open-source tools, such as Grid Nimbus.

It can be seen from the global high-performance computer TOP500 rankings that the architectures adopted by the high-performance computers on the rankings are mainly based on cluster technology and large-scale parallel processing technology. The high-performance computing cluster job processing architecture is shown in Figure 1. After the job is submitted, it is queued, assigned by the management node, and assigned to the computing node for processing. At present, high-performance computing usually adopts blade servers, and blade servers refer to high-availability, high-density, and low-cost server platforms [11]. Blade server refers to a standard-height rack-type chassis that can be plugged into multiple card-type server units. It is a low-cost server platform that realizes HAHD (High Availability High Density). It is a special application industry. And high-density computing environments are specially designed. Blade servers are like “blades,” and each “blade” is actually a system motherboard.

3.2. Parallel Computing. High-performance computing is an important branch of computer science, with the main goal of developing high-performance computers, researching parallel algorithms, and developing related software. High-performance computing is mainly measured by the speed of floating-point operations. Theoretical chemical calculations require an efficient and stable system environment. The availability of an environment for parallel computing and a job submission system directly affects the speed and quality of scientific research in theoretical chemical computing. High performance computers have floating-point performance.

Parallel computing is synonymous with supercomputing and high-performance computing, and is one of the important directions in the development of computer technology [12]. Figure 2 is a diagram of the parallel solution process of the problem.

3.2.1. Parallel Computing Architecture. Compared with serial computing, parallel computing is divided into time parallelism and space parallelism. Time parallelism is the

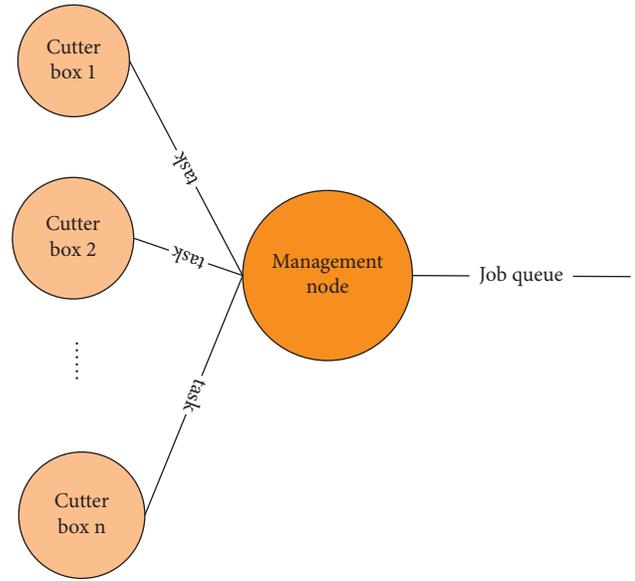


FIGURE 1: High-performance computing cluster job processing architecture diagram.

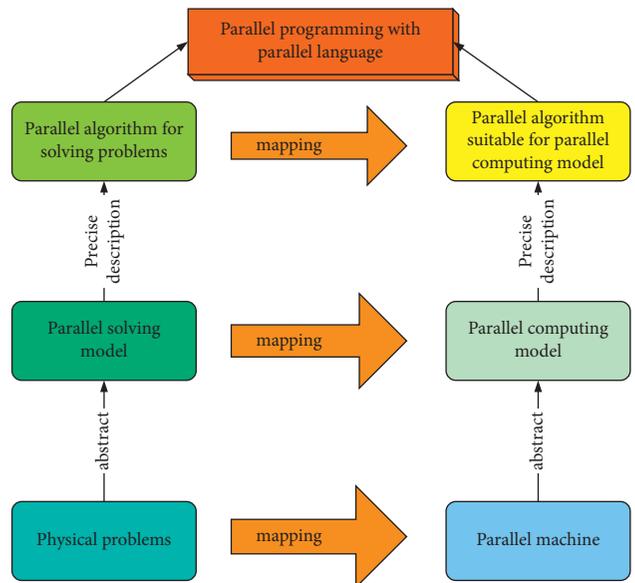


FIGURE 2: Diagram of the parallel solution process of the problem.

instruction pipeline technology. It decomposes the execution process of an instruction into several steps, and each step is completed by an independent component, thereby shortening the execution time of the entire task by executing

different instructions concurrently by independent components. Pipelining does not shorten the execution time of each instruction but improves performance by increasing the throughput of the microprocessor executing instructions. Whereas, spatially parallelism refers to the use of multiple processors or the use of multicore processors to perform computations concurrently. This paper mainly studies space-based parallelism. Common parallel architectures include SMP (symmetric multiprocessing), DSM (distributed shared memory), MPP (massively parallel processors), and cluster [13, 14]. The following comparison chart shows the differences between the four architectures, as shown in Figure 3.

3.2.2. Parallel Program Execution Time. Execution time refers to the time taken from the start of parallel program execution to the completion of all processes. It can be further divided into computation time, communication time, synchronization overhead time, and process idle time caused by synchronization. Computation time refers to the time spent by the process instruction execution, which can be divided into the time occupied by the program itself, namely the user time and the time spent by the operating system in order to maintain the execution of the program, that is, the system time [15]. The parallel execution time graph is shown in Figure 4.

3.2.3. Acceleration Coefficient. When measuring the performance of a multiprocessor system, an indicator usually used is called the acceleration factor, which is defined as follows:

$$V(h) = \frac{t_1}{t_h} \quad (1)$$

t_1 represents the best serial algorithm that uses a single processor to execute a serial program. Sometimes it also represents the execution time of a parallel algorithm on a single processor. These two representations are still different. t_h represents the execution time required for a parallel program to execute using h processors.

A parallel program can generally be divided into a serial part and a parallel part. If the second representation method is adopted, then t_1 and t_h can be expressed as follows:

$$\begin{aligned} t_1 &= g \cdot t_1 + (1 - g) \cdot t_1, \\ t_h &= g \cdot t_1 + (1 - g) \cdot \frac{t_1}{h}. \end{aligned} \quad (2)$$

Because:

$$V(h) = \frac{1}{(g + ((1 - g)/h))} \quad (3)$$

$$\frac{1}{(g + ((1 - g)/h))} = \frac{h}{g \cdot h + 1 - g}.$$

Then:

$$V(h) = \frac{h}{g \cdot h + 1 - g}. \quad (4)$$

Because:

$$\frac{h}{g \cdot h + 1 - g} \leq \frac{1}{g}. \quad (5)$$

So:

$$V(h) \leq \frac{1}{g}. \quad (6)$$

If the serial part of a parallel program accounts for 10%, that is, $g = 0.10$, according to this formula, no matter how many processors are used, the acceleration factor will be less than 10, which is the famous Amdahl formula.

3.2.4. Efficiency.

$$Effp = \frac{Vh}{h}. \quad (7)$$

Vh is the acceleration coefficient represented by formula (1), h represents the number of processor cores, and $Effp$ represents the performance-cost ratio obtained by using h processors or cores for parallel processing, which is generally less than 1. The value is an internationally common criterion for measuring the floating-point performance of high-performance computer systems. It is used to evaluate the floating-point performance of high-performance computers by testing the ability to solve linear algebraic equations.

3.3. GPU-Based Parallel Ant Colony Optimization Algorithm. In recent years, GPU (Graphics Processing Unit) parallel computing technology has become a research hotspot in the field of high-performance computing. GPU hardware has powerful floating-point computing capabilities, providing good support for large-scale scientific computing and engineering computing problems. Currently, in addition to traditional HPC applications, the demand for emerging HPC applications is also growing [16].

High-performance cloud computing can solve the user service problems faced in traditional high-performance computing. The hardware architecture of the GPU cluster is shown in Figure 5. Due to the common influence of various factors such as semiconductor technology, manufacturing technology, power consumption, etc., the current processor architecture presents a diversified development trend. Among them, GPU, as a kind of co-processor, has become one of the important components of contemporary high-performance computer systems and has developed rapidly. Its functions have developed from a single graphics display to high-speed parallel computing (General Purpose GPU, General Purpose GPU, GPGPU) in just ten years [17].

3.3.1. Basic Principles. Ant colonies, or more generally social insect colonies, are a distributed system. Although the individuals in the system are very simple, the whole system can

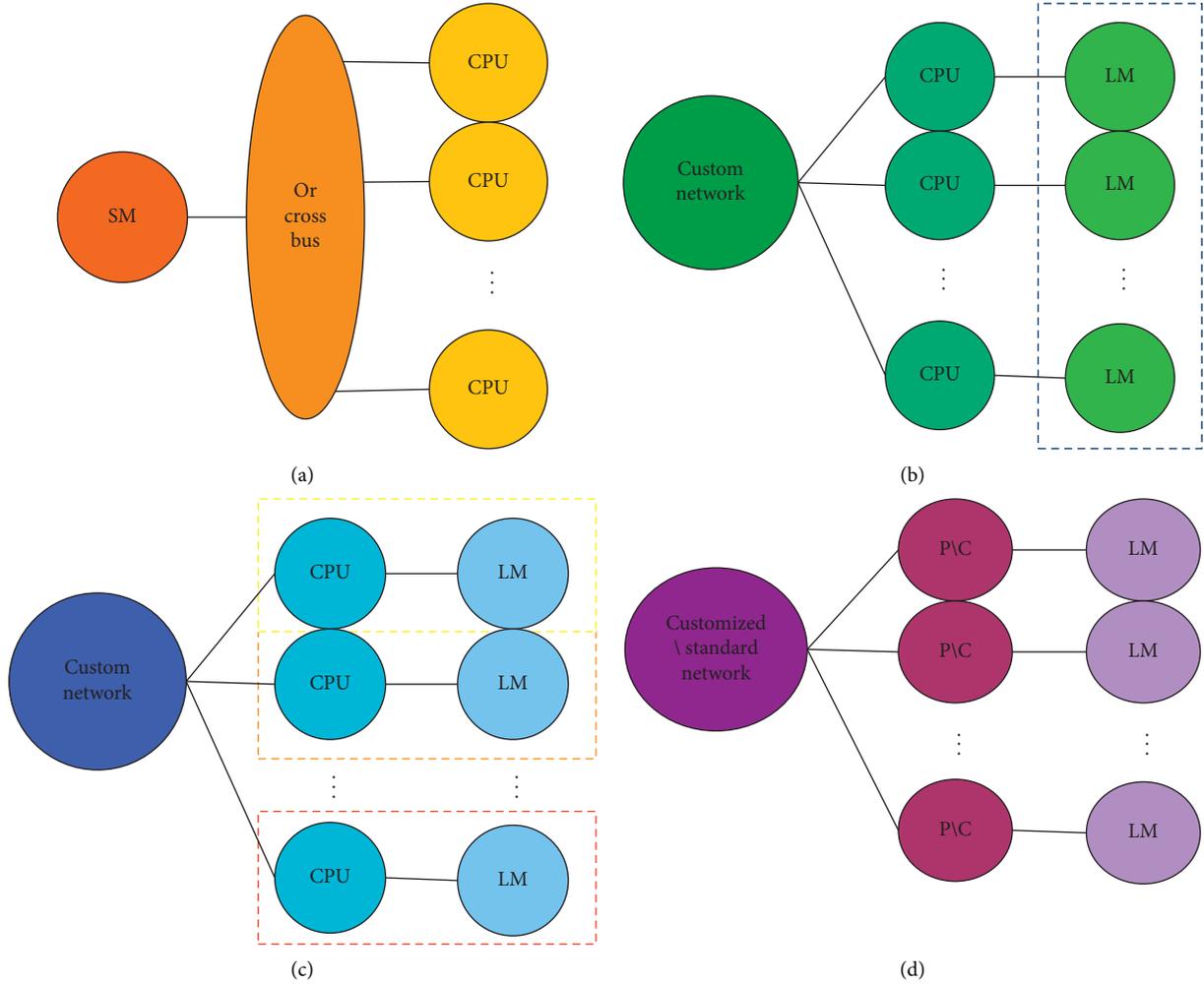


FIGURE 3: Four architecture diagrams. (a) SMP, physically single address space. (b) DSM, logically single address space. (c) MPP, physical/logically multiple address spaces. (d) Cluster, physical/logical multiple address space.

present a highly structured colony organization. Observations show that ants leave a secretion during their movement, and ants behind them can make a biased path choice based on the secretion left by the ants in front. This constitutes a positive feedback mechanism for learning information, and ants seek the shortest path to food through this information exchange [18, 19].

(1) Form description

$$GA = (P(0), N, L, Q, G, P, F, Z). \quad (8)$$

(2) Performance evaluation

The online performance is represented by the average value from the first generation to the current generation. Let $X_E(Q)$ be the linear performance of the strategy Q in the environment E , and $F_E(Z)$ be the objective function or the average fitness function corresponding to the environment E in the time Z or the Z th generation, then:

$$X_E(Q) = \frac{1}{Z} \sum_L^T F_E(Z). \quad (9)$$

The online performance represents the average value of the performance during the time period from the beginning of the algorithm to the current time, reflecting the dynamic performance of the algorithm.

Offline performance is the cumulative average of the best performance. Let $X_E^*(Q)$ be the linear performance of policy Q under environment E , then:

$$X_E^*(Q) = \frac{1}{Z} \sum_L^Z F_E^*(Z), \quad (10)$$

where

$$F_E^*(Z) = \text{BEST}(F_E^{(1)}, F_E^{(2)}, \dots, F_E^{(Z)}). \quad (11)$$

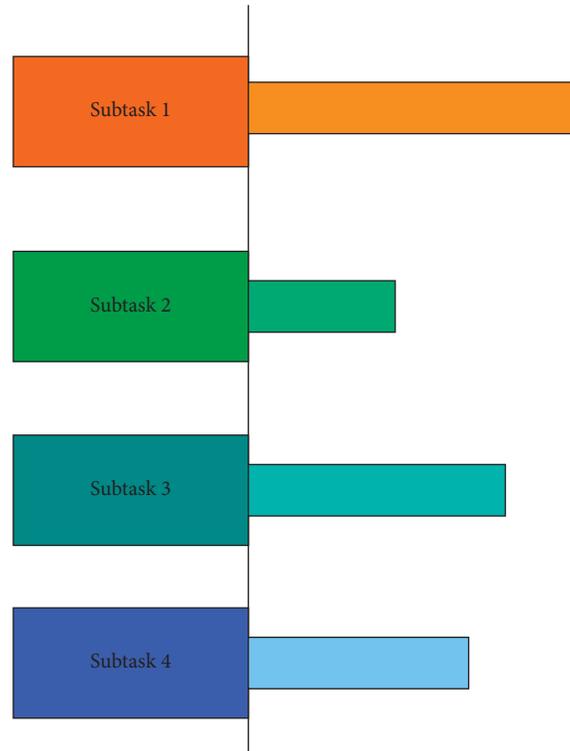


FIGURE 4: Parallel execution time graph.

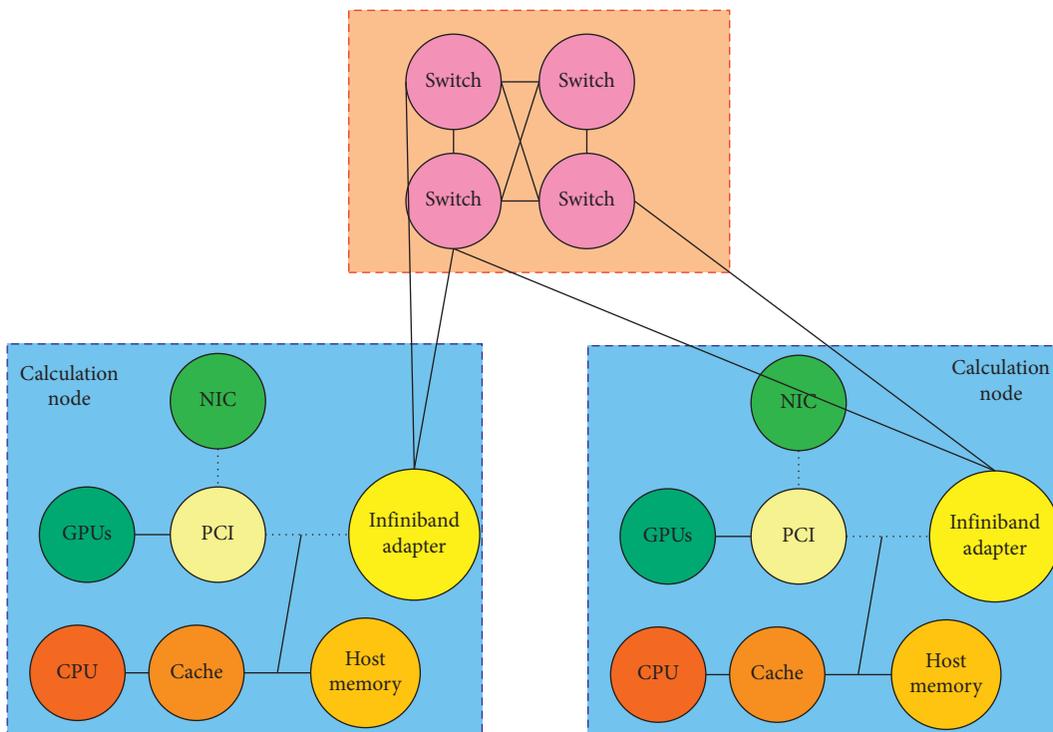


FIGURE 5: Hardware architecture of GPU cluster.

Offline performance represents the cumulative average of the best performance values of each evolutionary generation during the running of the algorithm, which reflects the convergence performance of the algorithm.

3.3.2. *Algorithm Process.* The process of ant colony algorithm optimization is actually controlled by three variables, namely state transition rule, pheromone local update rule, and pheromone global update rule.

The algorithm flow can be simply described as follows: each ant traverses all the cities according to the state transition rules and finds its own shortest path until all ants find their own solutions. Every time an iteration is completed, the pheromone on all paths is updated and the shortest path generated after this iteration is recorded until the termination condition is satisfied and the iteration ends [20]. In this process, the state transition probability can be defined as

$$P_{IJ}^K(Z) = \frac{\mu_{IJ}^a(Z)v_{IJ}^b(Z)}{\sum_{J=1}^N \mu_{IJ}^a(Z)v_{IJ}^b(Z)}. \quad (12)$$

Among them, v_{IJ} represents the visibility between the two places I and J , μ_{IJ} represents the concentration between the two places, a represents the importance of the pheromone concentration between the two places, and b represents the importance of the visibility between the two places.

Using the memory list to record the list of cities that ant K has walked through, the formula can be updated as

$$\mu_{IJ}(Z+1) = w\mu_{IJ}(Z) + \Delta\mu_{IJ}(Z+1),$$

$$\Delta\mu_{IJ}(Z+1) = \sum_{K=1}^M \Delta\mu_{IJ}^K(Z, Z+1), \quad (13)$$

$$\Delta\mu_{IJ}^K(Z, Z+1) = \begin{cases} \frac{O}{L_K}, \text{exit,} \\ 0. \end{cases}$$

Among them, $\Delta\mu_{IJ}^K(Z, Z+1)$ represents the amount of pheromone left on the path (I, J) by the movement of the K th ant at time $(Z, Z+1)$. $\Delta\mu_{IJ}(T, T+1)$ represents the amount of all ant pheromones in this process, and O represents the sum of all path pheromones. L_K represents the total length of the path taken by the K th ant, and w is the attenuation coefficient of the pheromone trajectory. Figures 6 and 7 are the standard path construction diagram and the path construction diagram of the prospect strategy, respectively.

4. English Language Feature Experiment and Analysis

4.1. Survey of the Corpus. This study extracted 160 article titles from ELT Journal and SSCI in 2020-2021 and used UAM Corpus version 2.0 to count them. Table 2 lists the basic information of the two journal corpora.

As shown in Table 2, the average length of sample titles in ELT Journal is 13.025 words, and the average length of sample titles in SSCI is 11.575 words. The average length of the two journal titles is basically the same, and both are around 12.300 words per title.

4.2. English Feature Structure Types. Based on some existing literature reference standards, this paper first divides the surface structure of sentences into the following four types.

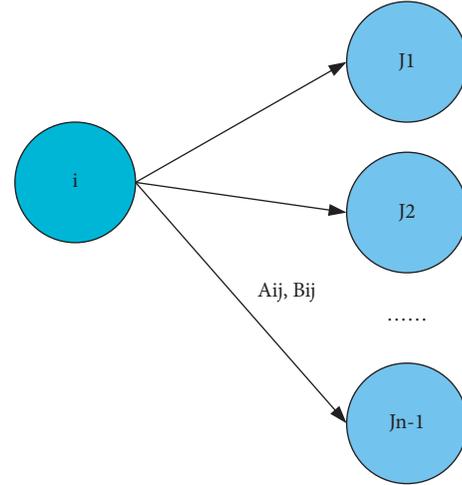


FIGURE 6: Standard path construction.

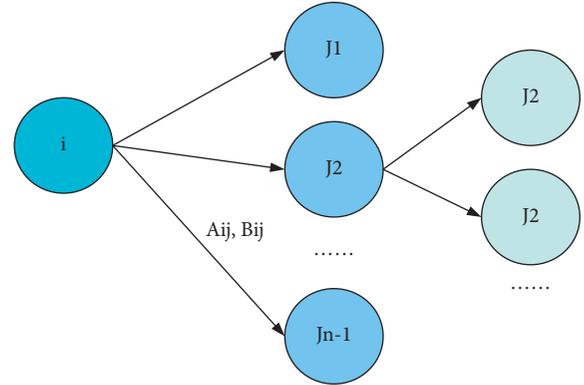


FIGURE 7: Path building for the outlook strategy.

Simple sentences are what we often call “subject-predicate structure” and “subject-predicate-object structure.” A simple sentence usually contains a main clause and a predicate, but sometimes a simple sentence also contains multiple subjects and predicates.

A coordinating compound sentence is a sentence consisting of two or three subordinate clauses. They are usually made up of coordinating connectives linked together. The main coordinating conjunctions are as follows: and, nor, but, or, yet, so, for, etc. In addition to using conjunctions to link, it can also use semicolons to link two parallel sentences.

A complex sentence is a sentence that consists of two or more subordinate clauses joined together using subordinating conjunctions. Common subordinating conjunctions are as follows: after, although, as, as if, as long as, as much as, as soon as, as though, because, before, even if, even though, once, until, when, since, so that, that, though, unless, in case, in order, whenever, where, wherever, what, whatever, etc. For example: I want to go to bed, because I am very tired.

TABLE 2: Basic information of ELT Journal and SSCI journal corpus.

Project	Number of titles	Number of words	Average title length (words/title)
ELT journal	80	1042	13.025
SSCI	80	926	11.575
Sum	160	1969	12.300

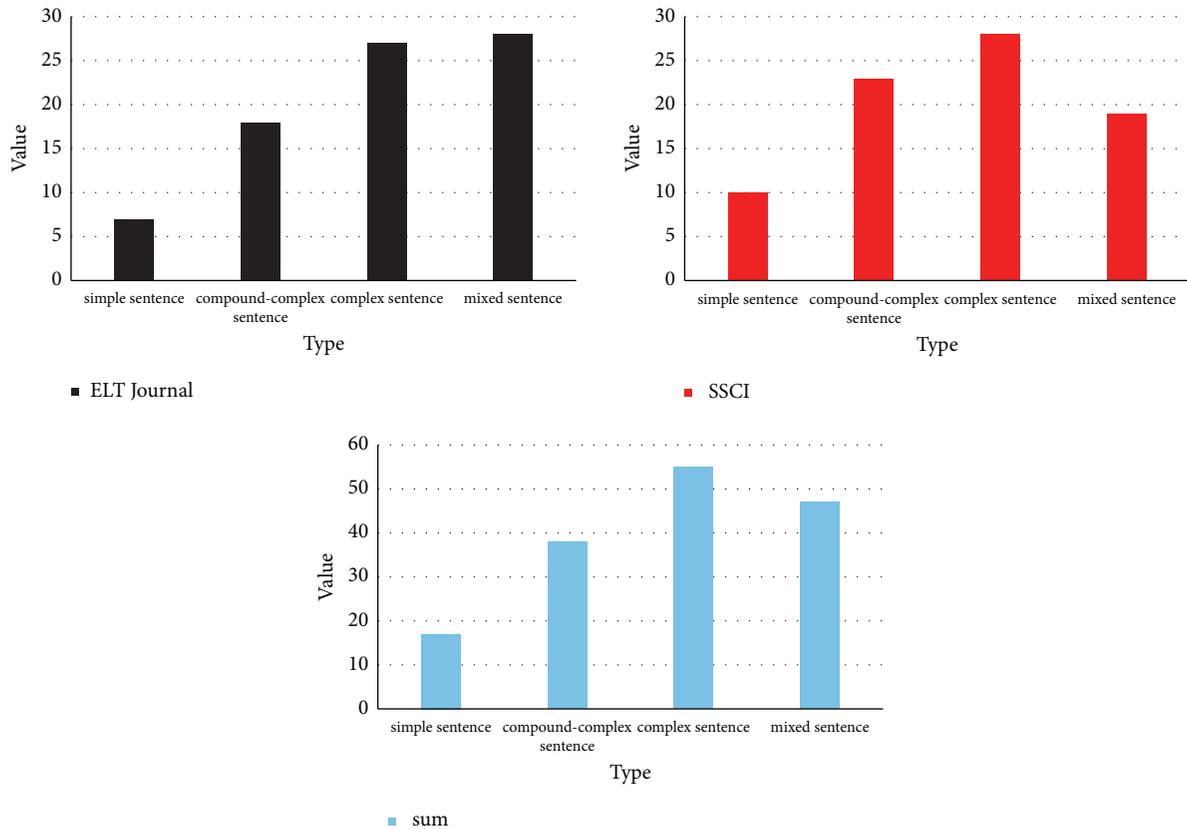


FIGURE 8: ELT Journal and SSCI English sentence structure types.

Simple sentences, coordinating compound sentences, and complex sentences can appear in a compound sentence. We often say that the subordinate clause is a typical mixed sentence. For example: the car, which is my father's, is in the garage.

4.3. Feature Analysis. It can be seen from Figure 8 that in ELT Journal, there are 7 simple sentences, accounting for 8.75%. There are 18 juxtaposed compound sentences, accounting for 22.50%, 27 complex sentences, accounting for 33.75%, and 28 mixed sentences, accounting for 35.00%. In SSCI, there are 10 simple sentences, accounting for 12.50%, 23 complex sentences, accounting for 28.75%, 28 complex sentences, accounting for 35.00%, and 19 mixed sentences, accounting for 23.75%.

Analysis of the data in Figure 8 shows that among the 160 English sentences in this article, complex sentences are the most used, with a total of 55 sentences, accounting for 34.38% of the sample. Complex sentences have a better connection between the upper and lower sentences, making the sentences more fluent and related.

In addition to the complex sentence result sentences mentioned, the second in this study is the sentence of mixed sentence type. The data collected by the author has a total of 160 sentences, of which 47 are mixed sentences, accounting for 29.38% of the sample. In terms of structure and function, mixed sentences can be regarded as an extension of simple sentences or as a simplified form of complex sentences. Generally speaking, it can convey or record more information with the help of premodifier or postmodifier through fewer words and simple structure. The distribution of mixed sentence structure types in the two journals is shown in Figure 9. For the convenience of expression, the simple sentence + coordinating compound sentence is expressed as SS + CCS, the simple sentence + complex sentence is expressed as SS + CS, and the coordinating compound sentence + complex sentence is expressed as CCS + CS. Simple sentences + coordinating compound sentences + complex sentences are expressed as SS + CCS + CS.

According to the analysis and as shown in Figure 9, the overall trend of ELT Journal and SSCI sentences is the same, and the three sentence structures (SS + CCS + CS) combine the most. Its frequency in ELT Journal and SSCI is 12 and 8

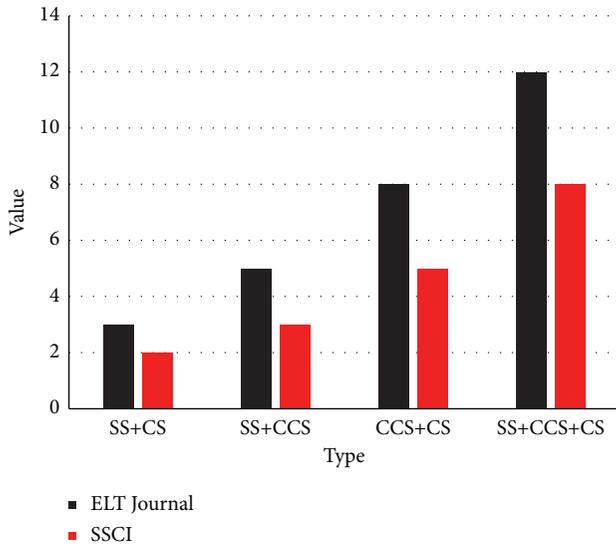


FIGURE 9: Distribution of mixed sentence structure types in ELT Journal and SSCI.

times, respectively, accounting for 15.00% and 10.00% of their respective corpora. Next is the statement of the compound sentence + complex sentence (CCS + CS) type, the highest proportion is 8 times and 5 times, accounting for 10% and 25%, respectively. From the point of view of collocation, the continuity of conjunction collocation is very strong. Different combinations of subordinate words and phrases can express different concepts, form different tense meanings, and convey different grammatical functions. Therefore, the structure of compound sentence + compound sentence (CCS + CS) helps scholars to provide research content and research methods more accurately.

5. Discussion

In recent years, GPU parallel computer technology has become a hot spot in the field of high-performance computers. Graphics hardware with powerful floating-point performance provides good support for large-scale scientific computing and engineering. At present, in addition to the traditional high-performance computer applications, the demand for high-performance computer applications is also increasing. In terms of user services, HPC faces many challenges: how to provide users with flexible services that allow users to independently manage data processing resources. How to save scalable dynamic computer resources and improve the utilization of high-performance computers [21].

With the deepening of research and the requirements of computing precision, the amount of data increases exponentially, which puts forward higher requirements for storage, computing, node communication, job allocation, and resource scheduling of high-performance computing clusters. Blindly adding hardware devices to improve computing performance not only brings huge power consumption but also has a bottleneck. Researching resource scheduling strategies to maximize efficiency has always been

the direction of high-performance development and the focus of scientific research.

Through the experimental analysis, it can be seen that the surface structure of sentences includes four types of structures: simple sentences, coordinating complex sentences, complex sentences, and mixed sentences. Based on the relevant corpus research, this paper mainly takes ELT Journal and SSCI as examples to discuss the structure and linguistic features of English sentences. From the statistical results, it can be concluded that the structure and linguistic features of English sentences in ELT Journal and SSCI are roughly equivalent. Among them, the mixed sentences appear the most, which are mainstream sentences, and in the mixed sentences, the sentence features that combine the three types of sentences account for the majority.

6. Conclusion

Language is a part of social culture, language problems are always closely linked with social problems, and all social, cultural and psychological factors limit the use of language. The results of this study show that although ELT Journal and SSCI cover a wide range of research areas, English sentences of the two journals share some commonalities in the structure and use of linguistic features. This is mainly reflected in the frequent use of mixed sentence structures. In terms of academic paper writing, the differences between Chinese scholars in spoken English and English writing require more in-depth comparative analysis and further academic research in the future [22]. It is hoped this article can help Chinese readers understand the use of international Frontier journal sentences in English linguistics and provide learners with a useful reference.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare no conflicts of interest. And author have seen the manuscript and approved to submit to your journal.

Acknowledgments

This work was supported by Guangxi Education Department in the research project entitled "Translation and Publicity Strategies of Guangxi Ethnomedicine" with No.2021ky0411.

References

- [1] R. P. Mundani, J. Frisch, V. Varduhn, and E. Rank, "A sliding window technique for interactive high-performance computing scenarios," *Advances in Engineering Software*, vol. 84, no. C, pp. 21–30, 2018.
- [2] W. Turek, "Erlang-based desynchronized urban traffic simulation for high-performance computing systems," *Future Generation Computer Systems*, vol. 79, no. 2, pp. 645–652, 2017.

- [3] S. Salimi, M. Mawlana, and A. Hammad, "Performance analysis of simulation-based optimization of construction projects using High Performance Computing," *Automation in Construction*, vol. 87, pp. 158–172, 2018.
- [4] K. E. Bouchard, J. B. Aimone, M. Chun et al., "International neuroscience initiatives through the lens of high-performance computing," *Computer*, vol. 51, no. 4, pp. 50–59, 2018.
- [5] S. Sharif, P. Watson, J. Taheri, S. Nepal, and A. Y. Zomaya, "Privacy-aware scheduling SaaS in high performance computing environments," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 4, pp. 1176–1188, 2017.
- [6] I. Noda, N. Ito, K. Izumi, H. Mizuta, T. Kamada, and H. Hattori, "Roadmap and research issues of multiagent social simulation using high-performance computing," *Journal of Computational Social Science*, vol. 1, no. 1, pp. 155–166, 2018.
- [7] L. Zhou, R. Islem, and C. Yan, "Special issue on high performance computing in bio-medical informatics," *Neuroinformatics*, vol. 16, no. 3-4, p. 1, 2018 G. Wu.
- [8] P. Thoman, K. Dichev, T. Heller et al., "A taxonomy of task-based parallel programming technologies for high-performance computing," *The Journal of Supercomputing*, vol. 74, no. 4, pp. 1422–1434, 2018.
- [9] M. Jiang, C. Bu, J. Zeng, Z. Du, and J. Xiao, "Applications and challenges of high performance computing in genomics," *CCF Transactions on High Performance Computing*, vol. 3, no. 4, pp. 344–352, 2021.
- [10] F. Iannone, G. Bracco, C. Cavazzoni et al., "MARCONI-FUSION: the new high performance computing facility for European nuclear fusion modelling," *Fusion Engineering and Design*, vol. 129, no. apr, pp. 354–358, 2017.
- [11] N. Jagiella, D. Rickert, F. J. Theis, and J. Hasenauer, "Parallelization and high-performance computing enables automated statistical inference of multi-scale models," *Cell Systems*, vol. 4, no. 2, pp. 194–206, 2017.
- [12] Z. Zhao, B. Guo, Y. Shang, and S. Huang, "Hierarchical and reconfigurable optical/electrical interconnection network for high-performance computing," *IEEE/OSA Journal of Optical Communications and Networking*, vol. 12, no. 3, pp. 50–61, 2020.
- [13] K. I. Volovich, S. A. Denisov, and S. I. Malkovsky, "formation of an individual modeling environment in a hybrid high-performance computing system," *Russian MicroElectronics*, vol. 49, no. 8, pp. 580–583, 2020.
- [14] A. Brinkmann, K. Mohror, W. Yu, P. Carns, and M. A. Vef, "Ad hoc file systems for high-performance computing," *Journal of Computer Science and Technology*, vol. 35, no. 1, pp. 4–26, 2020.
- [15] D. Langr, T. Dytrych, K. D. Launey, and J. P. Draayer, "Accelerating many-nucleon basis generation for high performance computing enabled ab initio nuclear structure studies," *Experimental Mechanics*, vol. 33, no. 3, pp. 522–533, 2019.
- [16] P. H. Tournier, I. Aliferis, M. Bonazzoli et al., "Microwave tomographic imaging of cerebrovascular accidents by using high-performance computing," *Parallel Computing*, vol. 85, no. JUL, pp. 88–97, 2019.
- [17] R. Diversi, A. Bartolini, and L. Benini, "Thermal model identification of computing nodes in high-performance computing systems," *IEEE Transactions on Industrial Electronics*, vol. 67, no. 9, pp. 7778–7788, 2019.
- [18] L. M. Ponce, W. D. Santos, W. Meira, D. Guedes, D. Lezzi, and R. M. Badia, "Upgrading a high performance computing environment for massive data processing," *Journal of Internet Services and Applications*, vol. 10, no. 1, pp. 1–18, 2019.
- [19] B. N. Chetverushkin, A. V. Saveliev, and V. I. Saveliev, "Kinetic algorithms for modeling conductive fluids flow on high-performance computing systems," *Doklady Mathematics*, vol. 100, no. 3, pp. 577–581, 2019.
- [20] Y. Shibuta, M. Ohno, and T. Takaki, "Computational metallurgy: advent of cross-scale modeling: high-performance computing of solidification and grain growth (adv. Theory simul. 9/2018)," *Advanced Theory and Simulations*, vol. 1, no. 9, Article ID 1870020, 2018.
- [21] A. Ahmad, A. Paul, S. Din, M. M. Rathore, G. S. Choi, and G. Jeon, "Multilevel data processing using parallel algorithms for analyzing big data in high-performance computing," *International Journal of Parallel Programming*, vol. 46, no. 3, pp. 508–527, 2018.
- [22] H. B. Celikoglu and J. Sanchez-Medina, "Special section editorial on \"High performance computing in simulation and optimization of dynamic transportation networks\" [guest editorial]," *IEEE Intelligent Transportation Systems Magazine*, vol. 10, no. 1, pp. 5–7, 2018.