行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※
※ ※
※ 分散式系統之總和數偵錯演算法之研究與設計 ※
※ ※
※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：■個別型計畫 □整合型計畫
計畫編號：NSC－89－2213－E－009－009
執行期間： 88 **年** 8 **月** 1 **日至** 89 **年** 7 **月** 31 **日**

計畫主持人：吳毅成
共同主持人：

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

執行單位：國立交通大學資訊工程系

中 華 民 國 89 **年** 8 **月** 31 **日**

行政院國家科學委員會專題研究計畫成果報告

分散式系統之總和數偵錯演算法之研究與設計

The study and design of detecting summative global predicates in distributed sustems

計畫編號：NSC 89-2213-E-009-009
執行期限：民國 88 年 8 月 1 日至民國 89 年 7 月 31 日

主持人：吳毅成　國立交通大學資訊工程系
計畫參與人員：李正軒　國立交通大學資訊工程系
蔡和諺　國立交通大學資訊工程系
陳俊琪　國立交通大學資訊工程系
鍾永良　國立交通大學資訊工程系

一、中文摘要

本計劃將研究有關分散式系統中，有關（token）總和數的偵錯問題。在分散式系統中，token 總和數的偵錯問題有很多應用。本計劃將有關 token 總和數的偵錯問題歸納為以下四種條件: 是否在某個狀態中，token 個數 (1)不等於 K；(2)小於 K；(3)大於 K；(4)等於 K，其中 K 為常數。

本計畫對這問題已作相當完整的分析，其成果如下：

對(1)項的每一次離線及線上偵測問題，我們證明針對分散式系統中的訊息，各別檢查即可。

對(2)(3)項的每一次離線偵測問題，我們解決的方法是將這些問題導為最大流量問題。藉此，這些問題都可在 O(n2 log n) 時間內解決，其中 n 是程式執行過程中的訊息數量。另外，本篇論文亦證明在時間複雜度方面，最大流量問題和(2)(3)項的偵測問題是一樣困難的。

對(2)(3)項的每一次線上偵測問題，我們解決的方法是將這些問題導為一個新的問題，稱為"線上最大流量問題"。本篇論文中，我們設計了新的線上最大流量演算法。藉此，(2)(3)項的每一次線上偵測問題都可在 O(n2 log n) 時間解決，與離線偵測演算法相同。也就是說，將離線演算法推廣成線上演算法的成本只有常數倍而已。事實上，我們的線上最大流量演算法對所有的流量網路皆可使用。假設流量網路有 m 線及 n 點。在 m=Q(n) 的情形下，新的線上最大流量演算法的時間複雜度，與目前已知最好的離線最大流量演算法的時間複雜度相同。在 m=O(n1+e) 的情形下，其中 e 是大於 0 之任意常數，我們的線上演算法的時間複雜度比目前最佳的離線最大流量演算法[1，19，35，36] 的時間複雜度只差 O(log n) 倍。

對(4)項的每一次離線及線上偵測問題，我們證明其為 NP-complete。

關鍵詞：分散式偵錯，分散式系統，全域條件，全域狀態

Abstract

In distributed programs, we usually keep some global predicates from being satisfied to make it easy to run the programs correctly. A common type of global predicates are: the total number of certain tokens in the whole distributed system is always the same or in specific ranges. In this project, we call this summative global predicates, classified into the following four: (1) at some global state of the system, N 1 K, (2) N < K (or N £ K), (3) N > K (or N 3 K), and (4) N = K, where N is the total number of tokens and K is a constant.

This project investigates the methods of detecting various summative global predicates. The first class of summative global predicates are trivial to detect by simply checking each message. For the second class of summative global predicates, Groselj, Garg and Chase solved the problem by reducing the problem to a maximum network flow problem. In this project, we propose an elegant technique, called normalization, to allow the second and third classes of summative global predicates to be solved by also reducing the problem to a maximum network flow problem. So, according to Goldberg and Tarjan's method, we can solve the problems in time O(n2 logn). For the fourth class of summative global predicates, we prove that it is a NP-complete problem.

In additional to reducing the second and third classes of summative global predicate problems to the maximum flow problem, this project also shows in a reverse manner that the maximum flow problem is also linear-time reducible to these predicate detection problems. Thus, we can conclude that the above summative global predicate problems are "as difficult as" the maximum flow problem in terms of time complexity.

Finally, in this project we design an incremental maximum flow algorithm that can be used to detect the second and third classes of summative global predicates incrementally. Interestingly, the time

complexity for the incremental algorithm is still O(n2logn), remaining the same. We also design an efficient algorithm for the incremental maximum flow problem by slightly modifying Goldberg and Tarjan's maximum flow algorithm. We find that the time complexity for the modified algorithm is still O(nm log(n2/m)), remaining the same, where m is the number of edges and n is the number of nodes in the flow network. In the case of m=O(n), our incremental algorithm has the same time complexity as those of the best current non-incremental algorithms. That is, the cost to pay for the incremental requirement is only a constant factor. In the case of m=O(n1+e) for any constant e>1, our time complexity is only O(log n) times higher than those of the best maximum flow algorithms by Cheriyan et. al. We leave these as open problems.

Keywords: distributed debugging, distributed system, global predicate, global state

二、緣由與目的

Error detection and debugging have been very important when programmers develop code. Most previous experiences and research reports showed that error detection and debugging are very time-consuming in a software development cycle [33]. This is because a bug may happen in an unexpected way at an unexpected spot. In single-processor systems, users usually debug programs by setting breakpoints in programs and then tracing the code step by step. Sometimes, programmers also put some assertions into the code in order to detect the correctness of the code.

With the rapid development of networks and distributed systems, programming on distributed environments is getting more common. However, the difficulty of distributed programming is much higher than that of sequential programming. Let us consider an example of debugging a distributed program on two processors. If we want to halt in a certain breakpoint of the program on one processor, it is very hard to halt the program on the other processor simultaneously. This makes distributed debugging very difficult.

Since distributed debugging is difficult, we detect errors in summative global predicates. The offline algorithms detect the global predicates after the execution of the program, while online algorithms detect the global predicates at each time when a new event happens during execution.

In distributed programs, we usually keep some global predicates from being satisfied to make it easy to run the programs correctly. A common type of global predicates are: the total number of certain tokens in the whole distributed system is always the same or in specific ranges. In this project, we call this summative global predicates, classified into the following four: (1) at some global state of the system, $N \neq K$, (2) $N < K$ (or $N \leq K$), (3) $N > K$ (or $N \geq K$), and (4) $N = K$, where N is the total number of tokens and K is a constant.

三、研究結果及與過去之比較

In our research, we consider both online and offline cases as follows.

Offline

The research results for the four classes of summative global predicates are listed as follows.

The first class of summative global predicates

For the first class of summative global predicates, it is trivial to detect the predicates by simply checking if each message is sent or received correctly. However, for other classes of summative global predicates, it becomes non-trivial. We need to keep track of all process states and then judge from all the states whether the global predicate holds. This makes the detection non-trivial.

The second and third classes of summative global predicates

For the second class of summative global predicates, recently, Groselj[17] proposed an interesting method to derive the snapshot with the minimum of the total numbers of tokens at all snapshots, called the minimum global snapshot [17], by reducing the detection problem to a maximum network flow (or minimum cut) problem. Later, Groselj, Chase, and Garg [4] developed the similar algorithm independently.

Although Groselj, Chase, and Garg can derive the minimum global snapshot, we find it non-trivial to reduce the above result to the maximum global snapshot for the third class of summative global predicates. This is because deriving a minimum network flow is an NP-complete problem [29], much more complex than deriving a maximum network flow.

For the third class of summative global predicates, we proposed the normalization technique in [6]. Then, based on this technique, we can easily detect the third class of summative global predicates in the same way.

From the above discussion, the time complexities of the minimum and maximum global snapshot problems will not be higher than those of the maximum flow problem. However, whether or not the time complexities for the above snapshot problems can be lower than those of the maximum flow problem remains unknown.

To resolve this question, this project shows in a reverse manner that the maximum flow problem is also linear-time reducible to these global snapshot problems. Thus, we can conclude that the above global snapshot problems are ``as difficult as'' the maximum flow problem in terms of time complexity.

The fourth class of summative global predicates

For the fourth class of summative global predicates, we prove in this project that it is an NP-complete problem.

Online

Consider the problem of online predicate detection. The research results for the four classes of summative global predicates are listed as follows.

The first class of summative global predicates

The first class of summative global predicates are trivial to detect by simply checking each message on-line.

The second and third classes of summative global predicates

For the second and third classes of summative global predicates, in this project, we design the on-line maximum flow algorithm that can be used to detect these summative global predicates in the on-line manner. The problem of finding a maximum flow in a directed graph with non-negative edge capacities has been a very important optimization problem in operations research and many other areas [11, 27].

Applications include transportation, communication, routing, graph partition, resource assignment, scheduling, and bipartite matching[27]. Researchers[1, 15, 16, 18, 34, 35] have investigated efficient algorithms for this problem for decades. In fact, it is also important to derive maximum flows on-line. Since the off-line minimum global snapshot problem can be reduced to the (off-line) maximum flow algorithm, the on-line minimum global snapshot problem can also be reduced to the on-line maximum flow algorithm.

A naive method to solve the on-line maximum flow problems is to use the best maximum flow algorithm to derive a maximum flow at each time when a new node and some edges are incorporated in a flow network. However, the time complexity for such a solution, in general, increases by a factor of O(n), where n is the number of nodes. The cost to pay for the on-line requirement is quite high. In this chapter, we design an efficient algorithm for the on-line maximum flow problem, based on Goldberg and Tarjan's maximum flow algorithm [16]. Interestingly, for all graphs, the time complexity for the algorithm is still $O(nm \log(n^2/m))$, where n is the number of vertices and m is the number of edges in the flow network.

In fact, our on-line maximum flow algorithm is general for all m (not limited to $m = n^2$). In the case of $m = n^2$, our on-line algorithm has the same time complexity as those of the best current off-line algorithms up to date.

In the case of $m = O(n^{1+e})$ for any constant e, our time complexity is only O(log n) times higher than those of the best maximum flow algorithms in [1, 18, 34, 35] up to date. Whether there are more efficient on-line algorithms in these cases are open problems.

For the second and third classes of summative global predicates, we can reduce the on-line detection problem to an on-line maximum flow problem with $m = n^2$. So, the time complexity for the on-line detection algorithms are still $O(n^2 \log n)$, the same as the off-line algorithms for the two classes.

The fourth class of summative global predicates

For the fourth class of summative global predicates, we show that detecting this class of predicates on-line is also an NP-complete problem.

四、計劃成果自評

本計劃對分散式系統中，研究有關 token **總和數的偵錯問題，含** online **及** offline**。本計劃將有關** token **總和數的偵錯問題歸納為以下四種條件**: **是否在某個狀態中，**token **個數** (1)**不等於** K**；**(2)**小於** K**；**(3)**大於** K**；**(4)**等於** K**，其中** K **為常數。對這問題已作相當完整的分析，成果如下：**

Offline **部份：對**(1)(2)(3)**項我們均找出最佳演算法。對**(1)(2)(3)**項問題，我們證明時間複雜度與最大流量問題同。對**(4)**項問題，我們證明其為** NP-Complete **問題。**

Online **部份：我們設計出一各極快的的演算法，與** Offline **演算法的效率同（註：一般而言，**Online **的演算法不應高於** Offline **的演算法, 因此，我們可稱其為最佳)。**

本計畫成果已發表於 Journals **有兩篇, 另外有兩篇正在審核中。**

五、參考文獻

[1]N. Alon. Generating pseudo-random permutations and maximum flow algorithms. Information Processing Letters, 35:201 204, 1990.

[2]L. Bouge. Repeated snapshots in distributed systems with synchronous communication and their implementation in CSP, 1987.

[3]K.M. Chandy and L. Lamport. Distributed snapshots: Determining global states of distributed systems. ACM Trans. Comput. Syst., 3(1):63 75, February 1985.

[4]C.M. Chase and K. Garg. Efficient detection of restricted classes of global predicates. In the International Workshop on Distributed Algorithms, September 1995.

[5]L.B. Chen and S.L. Lee. The parallel preflow push-pull algorithm on the maximum flow problem. Master thesis, Department of Computer Science and Information Engineering, National Cheng Chung University, July 1993.

[6]L.B. Chen and I.C. Wu. On detection of bounded global predicates. In Proceedings of the International Conference on Distributed Systems,

Software Engineering, and Database Systems. Taipei, 1996.

[7]Y.M. Wang P.Y. Chung and I.J. Lin. Checkpoint space reclamation for uncoordinated checkpointing in message-passing systems. Trans. Parallel and Distributed Systems, 8(6):165 169, June 1997.

[8]S.A. Cook. The complexity of theorem proving procedures. In Proceeding of Third Annual ACM Symposium on Theory of Computing, 1971.

[9]R. Cooper and K. Marzullo. Consistent detection of global predicates. Sigplan Notices, pages 167 174, 1991.

[10] L.R. Ford and D.R. Fulkerson. Maximal flow through a network. Siam Math., 8:399 404, 1956.

[11] L.R. Ford and D.R. Fulkerson. Flows in Networks. Princeton Univ. Press, Princeton, NJ, 1962.

[12] K. Garg and B. Waldecker. Detection of unstable predicates in distributed programs. In Proc. of the Conference on the Foundations of Software Technology and Theoretical Computer Science, December 1992.

[13] K. Garg and B. Waldecker. Detection of weak unstable predicates in distributed programs. Tran. Parallel and Distributed Systems, 5(3):299-307, March 1994.

[14] K. Garg and B. Waldecker. Detection of weak unstable predicates in distributed programs. Tran. Parallel and Distributed Systems, 5(3):299 307, March 1994.

[15] A. Goldberg. Recent Developments in Maximum low Problems. Technical report 98-045, NEC Research Institute, Inc., 1998.

[16] A. Goldberg and R.E. Tarjan. A new approach to the maximum-flow problem. Journal of the A , 35(4):921 940, October 1988.

[17] B. Groselj. Bounded and minimum global snapshots. Parallel and Distributed Technology, pages 72 83, November 1993.

[18] J. Cheriyan, T. Hagerup and K. Mehlhorn. An $O(n^3)$-time maximum flow algorithm. A Journal on computing, 25:1144 1170, 1996.

[19] R. Baldoni J.M. Helary and M. Raynal. Rollback-Dependency Rackbility: An optimal Characterization and its Protocol. University di Roma "La Sapienza", May 1997.

[20] D.R. Jefferson. Virtual time. A Transactions on Programming Languages and Systems, 7(3):404-425, July 1985.

[21] N. Plouzeau J.M. Heiary and M. Raynal. Computing Particular Snapshots in Distributed Systems. PhD thesis, Dept. of Elec. and Comput. Eng., Univ, of Texas, Austin., 1992.

[22] D.S. Johnson and K.A. Niemi. On knapsacks, partitions, and a new dynamic programming technique for trees. Mathematics of Operation Research, 8(1):1-14, 1983.

[23] E.N. Elnozahy D.B. Johnson and Y.W. Wang. A survey of rollback-recovery protocols in message-passing systems. Technical Report, 1996.

[24] L. Lamport. Time, clocks and the ordering of events in a distributed system. Communications of the ACM , 21(7):558 565, July 1978.

[25] L. Lamport. Time, clocks and the ordering of events in a distributed system. Communications of the ACM , 21(7):558 565, July 1978.

[26] Y.M. Wang A. Lowry and W.K. Fuchs. Consistent global checkpoints based on direct dependency tracking. Information Processing etters, 50(4):223 230, May 1994.

[27] R.K. Ahuja T.L. Magnanti and J.B. Orlin. Network Flows Theory, Algorithms, and Applications. Prentice-Hall, 1993.

[28] F. Mattern. Virtual time and global states of distributed systems. In Parallel and Distributed Algorithms Proceedings of the International Workshop on Parallel and Distributed Algorithms, pages 215 226. New York: Elsevier, 1988.

[29] D.S. Johnson M.R. Garey and L. Stockmeyer. Some simplified NP-complete graph problems. Theor. Comput. Sci., 1(1):237 267, 1976.

[30] H.B. Netzer and J. u. Necessary and sufficient conditions for consistent global snapshots. Tran. Parallel and Distributed Systems, 6(2):165-169, February 1995.

[31] H.B. Netzer and J. u. Finding consistent global checkpoints in a distributed computation. Tran. Parallel and Distributed Systems, 8(6):165 169, June 1997.

[32] J.M. Helary A. Mostefaoui R.H.B. Netzer and M. Raynal. Communication-Based Prevention of Useless Checkpoints in Distributed Computations. IRISA, Campus de Beaulieu, 35042 Rennes Cedex, FRANCE, May 1997.

[33] Shari Lawrence Pfleeger. The Production of Quality Software. Macmillan Publishing Company, 1991.

[34] S. Phillips and J. Westbrook. Online load balancing and network flow. In Proc. the Annual ACM Symposium on Theory of Computing, 1993.

[35] King S. Rao and R.E. Tarjan. A faster deterministic maximum flow algorithm. . Algorithms, 17:447 474, 1994.

[36] D.D. Sleator and R.E. Tarjan. A data structure for dynamic trees. Comput. Syst. Sci., 26:362 391, 1983.

[37] D.D. Sleator and R.E. Tarjan. Self-adjusting binary search trees. Journal of the ACM, 32:652-686, 1985.

[38] C.E. Leiserson T.H. Cormen and R.L. Rivest. Introduction to Algorithms. The MIT press, 1989.

[39] Y.M. Wang. Consistent global checkpoints that contain a given set of local checkpoints. Tran. Computers, 46(4):456 468, April 1997.

[40] I.C. Wu. Multilist Scheduling A New Parallel Programming Model. PhD thesis, School of Computer Science, Carnegie Mellon University, July 1993.

[41] J. Wu and R.H.B. Netzer. Adaptive independent checkpointing for reducing rollback propagation. In Proc. the Symp. on Parallel and Distributed Processing, pages 754-761, 1993.