A new statistical solution for the deadlock problem in resource management systems

By Z. GIDÓFALVI

1. Introduction

Nowadays the efficient resource management is an important problem of the national economy, mainly in the case of expensive resources. The resources handled by the operating systems of computers are expensive enough, therefore their optimum usage is an important problem.

The solution may be the proper choose of resource management strategies, on the other hand the economical avoidance of deadlock situations.

The deadlock state is a special, undesirable state of the resource management systems, in which some processes executed simultaneously in the computer get deadlocked by their requests, and there is no way to destroy this situation without external interference. An expressive example could be those processes (P_1 and P_2), which want to transmit data between two tape units (R_1 and R_2), P_1 from R_1 to R_2 and P_2 from R_2 to R_1. Assume a state in which P_1 holds R_1 and P_2 holds R_2, and both request the not acquired units. Then the requests will remain unsatisfied forever, because none of them can release the already acquired resources, and so P_1 and P_2 are deadlocked, and the system is in deadlock state.

In simpler systems both the detection and elimination of deadlock were the operator's duty, but in great systems this way seems to be rather unefficient. It is advisable to entrust this work to the computer, which can make a decision on the basis of theoretically established algorithms. Unfortunately the time requirement of procedures developed for detection and prevention is great enough, and thereby the efforts made for rentability cannot achieve the expected results. The new method described in this paper guaranties a smaller time complexity, and has some other advantages, too.

2. The system

The system can be described after [3] with the triple $\text{RMS}=\langle S, P, R\rangle$, where $S=\{S_1, \ldots, S_z\}$ is the set of system states, $P=\{P_1, \ldots, P_n\}$ is the set of processes, and $R=\{R_1, \ldots, R_m\}$ is the set of resources. Process P_i is a partial function

(P_i: $S \to 2^S$), because it can perform request, aquisition and release operations leading the system from state S_l into a set of states. Processes can interact explicitly, for example by exchanging messages, or implicitly, for example by competing for physical objects such as tape drives. Both types of interaction may cause the blocked state of processes, and they can be modelled by means of resources. The resources modelling the implicit interaction are called reusable resources, because after their request, acquisition and release by processes they are available again, and can be used in further cases. The resources modelling the explicit interaction are called consumable resources, because the message receiver process consumes (acquires) the resource produced (released) by the sender process after having requested it, and the consumed unit will never be available for other processes again. Moreover assume, that R_i has r_i units, where r_i is infinitely large for consumable resources.

The system states can be unambigously characterized by the number of resource units requested and acquired by processes, and can be illustrated by a directed bipartite graph [2, 3, 5, 6], where the nodes are from $P \cup R$, and the edges lead either from a P node to an R node or vice versa with the meaning:

1. $P_i \to R_j$ is a request edge indicating that P_i requests one unit from R_j;
2. $R_j \to P_i$ is an acquisition edge indicating that P_i holds one unit of R_j.

3. Deadlock strategies

If we do not influence the resource operations of P_i, and we have no information about the future requirement of P_i, then deadlock can always occur at the next step, which is to be detected and eliminated. The detection methods [1, 2, 3, 5, 6] decide, whether S_l is a deadlock state or not, and they are based on the following consideration: if there is a sequence, in which the processes can terminate their work one after the other from the state S_l — assuming that none of them will require more resources — S_l is not a deadlock state, because the former sequence is a possible state transition order. The time complexity of algorithms is not polinomial in general resource systems, where the resources are of both types. For reusable resources the algorithms require only mn steps, but the important consumable resources cannot be considered. The most complicated system, with mn time complexity of detection algorithm may have consumable resources, too, but with immediate allocation (all grantable requests to such resources are immediately fulfilled) [6]. There are efficient detection algorithms for restricted systems, too, but unfortunately with the same time complexity.

In the case of deadlock prevention we prohibit certain acquisition operations by means of information about the maximum claim of every P_i for all resources. We permit the acquisition in state S_l, when the maximum claim state $S_{l\max}$ (processes request their whole claim) is not a deadlock state, otherwise we prohibit this operation.

The deadlock avoidance is the simplest method for solving the deadlock problem. It means that the fulfillment of the necessary condition of deadlock should never allowed, since one can decide *a priori*, whether the system will reach a deadlock state or not. Often we use maximum claim information, too. The necessary condition of deadlock by [2, 3, 5] is the presence of at least one directed cycle in the graph representing the state. Later we will show, that there exist more efficient necessary conditions used in the new statistical method.

There are mixed solutions for the deadlock problem, which apply different strategies for different parts of the system [4, 6].

4. Necessary conditions of deadlock

Deadlock can occur due to one resource (reusable or consumable) and due to more resources. We will divide the necessary conditions accordingly.

Let R be a *reusable resource* with r units requested and/or acquired by the processes $P_1, \ldots, P_n$ with $p_1, \ldots, p_n$ units ($p_i \neq 0$). The necessary condition of deadlock on R is: $\sum_{i=1}^{n} p_i \geqq r+n$. To prove the expression assume, that the inequality is false. Then the number of edges directed to and from R is less than $r+n$. In the worst case all units of R are assigned to processes, and the remaining $n-1$ edges are requests. If we assume, that all the edges are directed from different processes to R (it is the worst case again), then there must be a process, say P_i, holding resources only. So we can find a sequence of processes beginning with P_i, in which all of the processes can terminate (not requiring more resources), and the state is deadlock-free.

Let R be a *consumable resource* produced (held) by $P_1, \ldots, P_n$. The necessary condition of deadlock on R is, that every process requests the units of R. To prove this statement assume, that there is at least one process producing R only. This process can produce an arbitrary number of units, thus all the others can terminate, and so the state is deadlock-free.

After the previous examinations we may consider at most two edges only between two arbitrary nodes, say P_i and R_j: $P_i \to R_j$ and $R_j \to P_i$. The necessary condition due to *more resources* is the presence of at least one directed cycle in the graph representing the examined state, including at least four nodes. To prove the former statement first we show, that a directed cycle in the graph is necessary. It is obvious, because without directed cycles in the graph we can make a queue from nodes, in which the edges from the k-th element are directed to the previous ones only. At the same time this order is a proper sequence of processes, in which they can terminate. Cycles appear through odd number of nodes only, but the cycles with two nodes are unimportant for us, because these ones were examined at the deadlock due to one resource. Therefore we can restrict the necessary condition for cycles including more than two nodes.

Contrary to [2, 5] the above necessary conditions permit directed cycles in the graph, when the conditions of deadlock due to one resource are satisfied.

5. The statistical method

Opposite to the former methods this strategy considers the system being determined by statistical information about its antecedents up to the moment of examination, and gives an approximate estimation about the existence of deadlock. We can assign a number v_i $(0 \leqq v_i \leqq 1)$ to every process and resource — the protection degree of P_i or R_i —, and w_i as the counterpart of v_i ($w_i = 1 - v_i$). The protection degree of a process or resource depends on its importance and value. Really w_i means the maximum allowed probability of coming the i-th node into a deadlock. The strategy examines the occurrence probability of deadlock on the basis of statistical information until this probability is smaller than the smallest w_i of all processes and resources concerned. If it is successful, then the deadlock occurrence probability is surely smaller than permitted, because it is surely smaller than the occurrence probability of necessary condition.

This strategy can be applied for detection and prevention, too, combined with detection algorithms, because deadlock can occur even if the deadlock occurrence probability is smaller than permitted, and naturally it is to be detected and recovered. At the application for detectional purposes we decide whether the prescribed protections can be satisfied or not — after the execution of a request or acquisition operation — in the future, too, and the exact detection algorithm is used accordingly. At the application for preventional purposes we examine the possible effects of acquisition before every acquisition operation, and we permit it, if the prescibed protections can be satisfied after the change, too. Naturally deadlock can occur in this case, too (with a small probability), thus the use of detection algorithms is necessary from time to time.

First of all let us see the statistical data needed for the decision. To gather them it is advisable to join the sampling of states with the resource operations, since in such cases there is a need for other administration activities, too. Let M be the operation counter with the initial value of zero, which is incremented by every operation. Moreover let $\mathbf{A}$ a hipermatrix of range $m \times n$, where a vector of five elements ($\mathbf{A}_{ij}$) belongs to R_i and P_j with the following meaning of coordinates:

1. k_{ji} — which shows the number of operations in which the graph had a request edge between P_j and R_i (the number of edges are irrelevant);
2. k_{jiu} — which shows, whether P_j had a request edge to R_i at the last operation between them ($k_{jiu} = 1$ if it had, otherwise 0);
3. b_{ji} — which shows the number of operations in which the graph had an acquisition edge between P_j and R_i (the number of edges are irrelevant);
4. b_{jiu} — which shows, whether R_i had an acquisition edge to P_j at the last operation between them ($b_{jiu} = 1$ if it had, otherwise 0);
5. M_{jiu} — which shows the value of M at the last rewriting of k_{ji} and b_{ji}.

This appearingly great amount of data makes possible to avoid updating all matrix elements at every operation (it would require $5mn$ steps). Rewriting of $\mathbf{A}_{ij}$ is needed only if P_j executes some kinds of operation with R_i or if we make use of the occurrence probability of the $R_i \to P_j$ and the $P_j \to R_i$ edges at the examination of the occurrence probability of necessary condition.

Thus k_{ji} gives the frequency of request occurrence, and b_{ji} gives the frequency of acquisition occurrence between P_j and R_i. The probability of these requests and acquisitions can be measured with their relative frequency:

the probability of $P_j \to R_i$ edge is k_{ji}/M, and

the probability of $R_i \to P_j$ edge is b_{ji}/M.

So we get a graph like the usual one representing the state, which characterizes the antecedents of the system up to the moment of examination, and whose edges are existing with the former probabilities.

We will divide the examinations about occurrence probability of necessary condition according to the previous chapter. So we introduce the numbers:

s_i — for the value $\sum_{j=1}^{n} p_j$ at the reusable resource R_i, and for the number of requesting processes at the consumable resource R_i,

q_i — for the number of occurrence frequency of necessary condition on the resource R_i (both types),

q_{iu} — which shows, whether the necessary condition was satisfied after the last request or release operation with R_i ($q_{iu}=1$ if it was, otherwise 0),

M_{iu} — which shows the value of M at the last change of q_i, and

$w_{i\min}$ — which is the smallest w_k of R_i or the P_j's connected to it (the connected processes have at least one edge to the reusable resource R_i, or they are producers of the consumable resource R_i).

With the above numbers we can formulate the feasibility of the prescribed protections. The protection degrees can be satisfied in the case of deadlock due to one resource, if the occurrence probability of necessary condition (q_i/M) is smaller than the smallest w_k of R_i and the processes concerned ($w_{i\min}$), thus $q_i/M < w_{i\min}$.

To the determination of deadlock probability due to more resources assume, that P_j is executing an operation with the resources of $R^* \subseteq R$. Then we try to build the graph, beginning with P_j through the resources of R^* in the case of request, and beginning with the resources of R^* through P_j in the case of acquisition, i.e., we try to put the nodes into levels from the root (s) through the directly, secondarily etc. accessible nodes, where the k-th level consists of such resources or processes, to which there is an edge directed from at least one node of the $(k-1)$-th level. Thereby every node on the k-th level is accessible from the root (s) (P_j or R^*). Executing the request or acquisition operation the edges between the first and second level are existing surely, and our purpose is to examine the probability of a directed cycle in the graph, due to the newly introduced edges. Assume that after a request operation of P_j, P_j appears once more say on the k-th level. This means a directed cycle, which includes the nodes $P_j, a_1, \ldots, a_{k-2}, P_j$. Mentioned previously the probability of $P_j \to a_1$ edge is considered to be 1, and the probability of a certain, say $a_n \to a_{n+1}$, edge in the cycle is $k_{n,n+1}/M$ or $b_{n,n+1}/M$ depending on the type of the edge (request or acquisition). Since the processes perform their request and acquisition operations in a nondeterministic manner, the existence of any two edges are independent events, and so the probability of any cycle in the graph can be obtained by multiplying the occurrence probabilities of the edges in the cycle. The correlation between the cycles are ignored and so the access probability of a node accessible through more than one paths is substituted with the highest probability of paths.

But the complete tracing of the cycles is unnecessary, because we are interested in the feasibility of the prescribed protections, and not in the exact probability of a cycle. If there is an a_i in the former example, for which the probability of getting from P_j to a_i is smaller than the smallest w_k of nodes on the mentioned path, then the further examination of this cycle can be ignored, since the prescribed protection degrees are already satisfied on this path. For the sake of simplicity the existence of low probability parallel paths is neglected in this algorithm.

Summarizing we must continue the cycle detection from P_j until:

1. for all paths from P_j the condition mentioned above is fulfilled, and so we can satisfy the protection degrees, or until

2. P_j occurs once more in the graph, and the condition has not been fulfilled yet, so in this case we cannot satisfy the protection degrees for the nodes in the cycle.

In the case of an acquisition operation executed by P_j the graph building and the examinations can be made similarly.

The time complexity of the algorithm ([6]) performing the above examinations is less than mn, since the examination of feasibility on one resource requires one step only (altogether m steps), and the number of steps needed at the cycle detection is surely less than mn, because not all the $2mn$ edges are considered. The other essential advantage of the strategy is the option of giving different protection degrees to processes and resources. Thus it is possible to rise the protection degree of a process as a function of the performed work.

The strategy can be mixed with others, too. Thus it is easy to apply prevention or avoidance methods instead of statistical ones on singular resources. This is the case on every R_i, where $w_{i\,\min}=0$.

DEPT. OF MATHEMATICS
ELECTRICAL ENGINEERING FACULTY
TECHNICAL UNIVERSITY BUDAPEST
STOCZEK U. 2./H.
BUDAPEST, HUNGARY
H—1111

References

[1] COLDEWEY, H. D., J. GONSCHOREK, H. HOECHNE, E. SCHÖNBERGER, F. STETTER, Ein Modell zur Deadlockerkennung, *Ang. Informatik*, v. 17, 1975, pp. 65—69.

[2] HOLT, R. C., On deadlock in computer systems, Ph. D. thesis, Cornell Univ. Ithaca N. Y., 1971.

[3] HOLT, R. C., Some deadlock properties of computer systems, *Comput. Surveys*, v. 4, 1972, pp. 179—196.

[4] HOWARD, J. H., Mixed solution for the deadlock problem, *Comm. ACM*, v. 16, 1973, pp. 427—430.

[5] SHAW, A. C., *The logical design of operating systems*, Prentice Hall, 1974.

[6] GIDÓFALVI, Z., *Erőforrásgazdálkodó rendszerek pattállapotai és feloldásuk*, dissertatio, BME, 1978.

(Received Sept. 2, 1978)