Proceedings of the First International Conference on Machine Learning and Cybernetics, Beijing, 4-5 November 2002

USING GASA TO SOLVE DISTRIBUTED REAL-TIME SCHEDULING PROBLEMS

WEI PU, YI-REN ZOU

ERC of Integrated Automatic Technology, Institute of Automation Chinese Academy of Sciences,
Beijing, 100080, P.R.China
E-MAIL: pv@sample.ia.ac.cn, zyr@sample.ia.ac.cn

Abstract:

In addition to constraints of timing and resources, the constraints of precedence must be considered in the distributed real-time system. Scheduling problem is an NP-complete problem. The paper presents scheduling model based on precedence graph. According to this model, we design the rules of coding, decoding and fitness function etc, and construct a scheduling table that meets all constraints with genetic algorithm and simulated annealing (GASA). With this table, distributed real-time systems can schedule tasks without jittery. The example of FF system is given to illustrate the effectiveness and feasibility of our proposed algorithm.

Keywords:

Distributed real-time system; Precedence graph; Scheduling table; Genetic algorithm and simulated anealing (GASA)

1 Introduction

Distributed real-time systems are becoming more and more commonplace. The systems are defined as those systems in which the correctness of the system depends not only on the logical result of computation, but also on the time at which the results are produced. [1]. The basic problem in the real-time systems is to make sure that tasks must meet their time constraints. A system, which assigns tasks on-line in the uncertain environment, can't meet all tasks' deadlines. In order to meet the deadlines of critical tasks in the worst condition, the priorities of tasks must be pre-allocated. In general, there are periodic and sporadic tasks, which can be transformed into periodic tasks in the system. The primary criterion in the static scheduling of periodic tasks is predictability determining a feasible schedule wherein all tasks must meet their timing, precedence and resource constraints.

Given a set of periodic tasks and their characteristics, a typical algorithm that deals with a distributed system attempts to assign instances of the tasks to processors in the system and to construct a scheduling table, which identifies the start and completion times of all instances in the periodic tasks. At run-time, the set of tasks is repeatedly executed according to this scheduling table every macro-cycle. Macro-cycle is an important parameter associated with scheduling table and the LCM (the least common multiple) of the periods. [2]. For periodic tasks, there exists a feasible schedule if and only if there exists a feasible schedule for the macro-cycle. Each task has $marcocycle/T_i$ instances in the macro-cycle. With the period, start and finishing times of the first instance in the task, scheduler can dispatch this task without jitteriness, so the memory requirements will be deceased.

2 Distributed real-time scheduling model

In general, to define a scheduling problem of the distributed real-time system, we need to specify two sets: a set of n tasks $J=\{J_1,J_2,....J_n\}$ and a set of m processors $P=\{P_1,P_2,...P_m\}$. Each task $J_i(P_k)$, which can be characterized by three parameters (T_i,C_i,D_i), must be executed on the processor P_k, where T_i,C_i,D_i represent the period, worst case execution time and deadline of task J_i respectively. Moreover, precedence relations among task can be specified through a precedence graph.

In the certain real-time applications, tasks can not be executed in arbitrary order but have to respect some precedence relations defined at the design stage. The notation $J_1 \rightarrow J_2$ specifies that task J_1 is an immediate predecessor of J_2, meaning that graph contains an arc directed from node J_1 to node J_2.

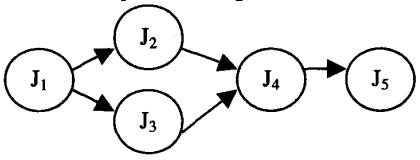


Fig. 1. Example of a precedence graph

Fig. 1 illustrates a precedence graph that describes the precedence constraints among five tasks. As J_1 is completed, either J_2 or J_3 can start. Task J_4 can start only when J_2 and J_3 are completed. Such precedence

0-7803-7508-4/02/$17.00 ©2002 IEEE

relations are usually described through a directed acyclic graph

$$G = \{J_i \rightarrow J_k\} = \{(J_i, J_k)\}.$$

The distributed real-time system can be described as the scheduling model

$$\{J, P, G, C, D\},$$

Where $C = \{C_1, C_2, ..., C_n\}$,

$$D = \{D_1, D_2, ...D_n\}.$$

If the deadlines of tasks are equal to their periods, then this model is converted to $\{J, P, G, C, T\}$. In this model, P represents the resource requirements in the system, G does the precedence relations and C, T does the timing constraints.

Given the scheduling model, the schedule can be formally defined as a function

$$S(J, P, G, C, T),$$

Schedule means to assign processors from P to tasks from J in order to complete all tasks under the imposed constraints.

3 Scheduling method

If the tasks have simple characteristics, then a table can be constructed by using the earliest-deadline-first (EDF), or the rate-monotonic (RM) scheduling algorithm. [3][4]. However, besides timing constraints, tasks may have resource requirements and can possess precedence. In the cases, scheduling problem is an NP-complete problem, and the present of complex precedence constrains exacerbates the problem. [5]. Hence, heuristics and approximation algorithms, i.e., brand-and-bound and simulated annealing, are resorted to deal with this problem.

In order to improve the performance of optimization, we adopt genetic and simulated annealing (GASA) in searching for a feasible schedule. Now the detail design of hybrid genetic algorithm is shown as following: [6].

1) Design of coding

Each gene in the chromosome is denoted by $J_i \times 10^1 + P_k \times 10^0$. The chromosome is denoted by jobs' execution sequence, which is a natural expression of scheduling problems.

2) Design of decoding

Due to precedence relations and resource requirements, decoding must be designed in detail. The current gene is decoded if there don't exist its immediate predecessors in the chromosome, or the next gene will be decoded. After decoded, this gene will be made a remark. We are able to know the existence of the current gene's unremarked immediate predecessors according to precedence graph. Average computational complexity is $O(m,n) = m \times n/2$, where $m = \dim(G)$ and $n = \dim(J)$. The maximum $f_{i,1}$ in the all-immediate predecessors is the start time t_{start} of the current task. According to the period and execution time of the current task, the time slices of all instances are generated as follows:

$$\cup[kT_i + t_{start}, kT_i + t_{start} + C_i]$$

Where $k \in \{0,1,..., macrocycle/T_i - 1\}$

We must check whether these time slices are collided with others, and adjust t_{start}, and repeat these steps before all genes are decoded.

3) Method for generating initial population

The initial population consists of N chromosomes, which are generated by the random combination from one feasible schedule.

4) Selection operator and definition of fitness function

We adopt roulette way as s selecting method. It's a fitness proportional selecting strategy. In order to prevent the optimal solution from being destroyed by crossover and mutation operation, we also use the elitist way on the basis of roulette way that can accelerate the searching speed. The goal of optimization is to minimize the number of late tasks N_{late},

$$N_{late} = \sum_{i=1}^{n} miss(f_i)$$

Where $miss(f_i) = \begin{cases} 0 & if\ f_i \le d_i \\ 1 & otherwise \end{cases}$,

d_i is the deadline of the i^{th} task, f_i is the finishing time of the i^{th} task, so we can define fitness function as follows.

$$f(S_i) = e^{-N_{latei}}$$

Where S_i is the i^{th} chromosome in the population.

5) Crossover operator

We adopt one-point crossover. Firstly, selecting two parent chromosomes in the population randomly; Secondly, selecting a breakpoint randomly; Thirdly, selecting the part behind the breakpoint of the first parent chromosome as a part of the child chromosome; Finally, selecting legal genes of the second chromosome to fill the remnant part of the child chromosome. By these operations, we can produce two legal child chromosomes from the parents and guarantee schedule feasibility. For example:

Parent 1: 1 2 3 4 5
Parent 2: 3 5 2 1 4
Suppose the breakpoint is 3, then
Child1: 2 3 5 1 4
Child1: 3 2 1 4 5

6) Mutation operator

We adopt exchange mutation. The genes in the parent chromosome are selected randomly and exchanged each other. For example:

Parent: 1 2 3 4 5

Suppose two positions of selected genes are 2 and 4, then

Child: 1 4 3 2 5

7) Simulated annealing

Simulated annealing is a local search method, which seeks to avoid being trapped in s local minimum. A chromosome S is chosen randomly from the population after crossover and mutation operation. According to S, a new chromosome S' can be generated randomly, and be accepted if

$$random[0,1] < \min\{1, \exp(-\frac{f(s')-f(s)}{t_k})\} \quad ,$$

where $t_k = \lambda t_{k-1}\ (0<\lambda<1)$ is a sequence of positive control parameters with $\lim_{i\to\infty} t_k = 0$. If $f(S') \le f(S)$ then S is replaced by S' with probability one. If, on the other hand, $f(S') > f(S)$ then S is replaced by S' with some probability. This probability decreases with increasing k.

8) Rule of termination

Use the preset maximum generation $N_{\max}$ or $N_{late}=0$ as termination condition.

4 Experimental results

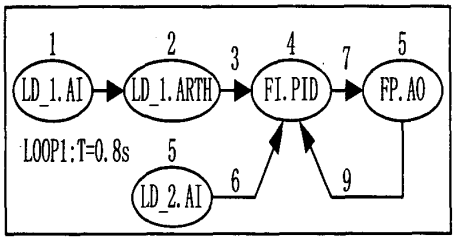
Foundation fieldbus (FF) control system is a type of distributed real-time system. The execution and communication of function blocks distributing in the different devices must meet the constraints of timing, precedence and resource. [7]. In fact, the real-time scheduling includes scheduling of function blocks and bus. [8]. There are six devices in the single link as shown in the table 1.

Table 1. the list of devices

Index	1	2	3	4	5	3
Devices	LD_1	LD_2	TT	FY	FI	FB

The control configuration is made up of ratio and split range control, as shown in the Fig. 2. There are two control loops, six devices and function blocks, and we can see precedence relations G among function blocks. For example,

$$G = \{(1,2),(2,3)...\}$$



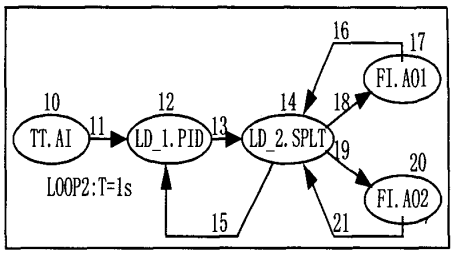


Fig.2. Control configuration graph

Genetic algorithm and simulate annealing are used to solve the problem. The parameters of GASA algorithm are given as follows: the population size N=50, the crossover rate $P_c = 0.6$, the mutation rate $P_m = 0.1$, the annealing rate $\lambda = 0.9$, and the maximum generation $N_{\max} = 25$. The optimal scheduling result as shown in the Fig. 3:

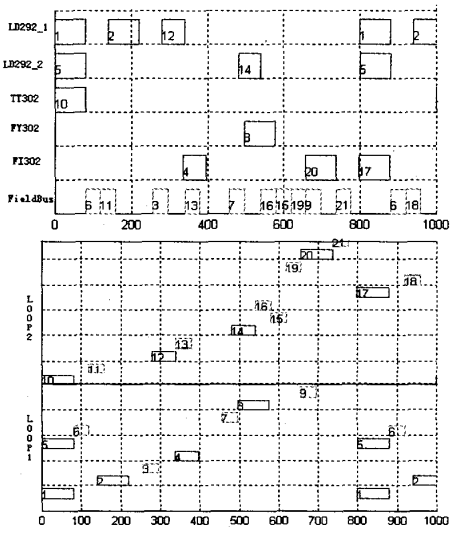


Fig.3. Gantt charts

In the Fig.3, Gantt charts are device oriented and control loop oriented. The vertical axis are defined as devices and control loops respectively, the horizontal axis represents time. Blocks represent the time slices of function blocks execution. For example, "1" in the block represents LD_1.AI because its index is "1".

5 Conclusions

The paper makes a deep study to distributed real-time scheduling problem, considers the constraints of timing, resource and special precedence, and constructs scheduling model based on precedence graph. GASA is used to search the optimal schedule. Finally, scheduling example in the FF system illustrates the effectiveness of our proposed method

that provides a new way to study scheduling problems in the distributed real-time system.

References

[1] J.A.Stankovic, Misconceptions About Real-time Computing. IEEE Computer, Vol. 21, No. 10, 1988

[2] E.L.Lawler and C.U.Martel. Scheduling Occurring Tasks on Multiple Processors. Information Processing Letter, 12(1), 1991

[3] C.L Liu and J.W. Layland (1973) "Scheduling Algorithms for Multi programming in a Hard Real-time Environment," Journal of the ACM

[4] Ramamritham, G.Fohler and J.M.Adan, Issues in the Static Allocation and Scheduling of Complex Periodic Tasks, 10th IEEE Workshop on Real-time Operating Systems and Software, May 1993

[5] A.K.Mok, Fundamental Design Problems of Distributed System for Hard Real Time Environments, Ph.D. Thesis, Lab for CS (MIT), TR-297, 1983

[6] Wang Dingwei, Tang Jiafu, Huang Min. Genetic Algorithm and EngineeringDesign. Beijing: Science Press, 2000

[7] "FOUNDATION™ Specification System Architecture", Revision 1.4, Fieldbus Foundation, 1999

[8] "FOUNDATION™ Specification Data Link Protocol Specification", Revision 1.4, Fieldbus Foundation, 1999