The Journal of Natural Science,
Pai Chai University, Korea
Volume 7: 115~120, 1995

A Theoretic Foundation of Imprecise Computation

DongWon Park
Department of Computer Science
PaiChai University

근사값 계산 방안에 대한 이론적 연구

박동원
배재대학교 전자계산학과

Imprecise computation has been suggested as a promising model of real-time computing in order to deal with timing constraints imposed by the environment. However, the theoretical foundation of the technique has not been fully explored. To address this, a decision-theoretic foundation of imprecise computation is proposed. The main benefit of such a treatment is that it enables the qualitative assumptions underlying imprecise computation techniques to be explicitly stated in a formal way. The theoretical foundation laid out in this paper, hence, will not only enable the justification of using imprecise computation techniques for a real-time application, but will also facilitate the development of extended techniques for more complex real-time systems.

실 시간 처리 시스템에서의 시간제약 조건을 충족시키는 바람직한 수단으로 근사값 계산 방안이 제안되었지만, 이에 대한 근본적 이론연구는 깊이 고찰되어 있지 않은 형편이다. 이에 본 논문에서는 실제 응용 프로그램에서의 근사값 계산 방안의 채택 당위성을 판단할 수 있는 이론적 결정 방법을 제시하였다. 이로써 실시간에서의 시스템 예측도와 대응능력을 높여, 효과적인 자원관리와 제약조건을 충족시킬 수 있도록 한다.

Keyword : Real Time, Imprecise Computation, Anytime Algorithm, Resource Allocation, Timing Constraints

I. Introduction

A real-time system is a computer system that operates under externally imposed resource constraints which must be satisfied if correct operation is to be achieved. The resource constraints imposed by the environment raises requirements and issues that are not usually addressed in traditional systems. In particular, neither producing correct result nor being "fast" is in itself sufficient to be real-time. Real-time systems must be able to perform their computations while satisfying resource constraints in a predictable fashion.

An approach that has been proposed

for obtaining the desired predictability is the notion of imprecise computation[1,2] (a similar notion, anytime algorithms[3,4,5], has also been developed in the real-time AI community). The idea of imprecise computation is that if we do not have sufficient resources to complete a perfect computation, we perform only a part of it and produce some approximate results. The focus is then on maximizing the quality of this paper result and ensuring that it meets certain acceptability criteria. Work in this area has been in the general context of real-time systems, and on designing algorithms which can produce useful partial results.

The main objective of this paper is to explore the theoretical foundation of imprecise computation. Investigating the theoretical foundation of imprecise computation is important for two reasons. First, it can improve our understanding about the applicability of the technique. Second, it can help us to identify potential extensions to the technique.

Ⅱ. A Decision-theoretic View of Imprecise Computation

There are two fundamental ideas behind imprecise computation. First, a program for solving a problem under external timing constraint is designed such that it consists of a series of steps, each of which generates a partial result at the end of each step. A computation step in this context refers to a piece of the algorithm that is atomic in the sense that it can not be further decomposed into smaller steps that generate some partial results. Second, computation times are allocated to these atomic steps dynamically based on two techniques: reactive imprecision and predictive imprecision.

The reactive imprecision techniques[1] (also called the milestone technique) simply allocates computation time to atomic steps in a predefined order. The atomic steps are usually ordered in a way such that an acceptable partial result can be generated as early as possible(i.e., using minimal resource). The advantage of reactive imprecision is that it ensures that a minimally acceptable result can be produced using minimal resources. Predictive imprecision, which was also referred to as the sieve technique in the paper[1] can be used if the computation is structured as a series of mutually independent steps, and an estimate of the amount of time required for the execution of each step is available. The system scheduler compares the time needed for the computation with the time available, and if it is insufficient, choose some steps to be skipped so that the rest of the computation may complete in time. This technique requires more system support than reactive imprecision does, for it requires a tool to assess execution times for each step and a scheduler to choose which steps to skip. It is a predictive technique in that it requires precise prior knowledge of the timing characteristics of the computation and their time available, and cannot respond to dynamic changes in these. In this section, we will show that imprecise computation techniques can be viewed as partially compiled decision-theoretic resource allocation strategies based on qualitative information regarding resource requirements and result quality.

1. The Foundation of Reactive Imprecision

Let us consider a simple problem of

allocating computational resources R among two computation steps a_1 and a_2. Depending on the resources allocated, each computation step may or may not complete (denoted by C). If the computation step completes, it may or may not be successful (denoted by S) in generating a result. In our discussion, we will use CS_i to denote that computation step a_i completes and is successful; $\neg CS_i$ to denote that a_i does not complete or fails to generate a result. If the application is such that the consequence of not producing any result could be disastrous(the objective will not be attained), then we have a large negative utility associated with total failure. Under this circumstance, we can assume that the utilities of possible outcomes be the following:

- $CS_1 \wedge CS_2$: utility = $u_1 + u_2$
- $\neg CS_1 \wedge CS_2$: utility = u_2
- $CS_1 \wedge \neg CS_2$: utility = u_1
- $\neg CS_1 \wedge \neg CS_2$: utility = -N

where N is much larger than u_1 and u_2. let the probability that the first step can complete be p_1, and the probability that the second step can be complete be p_2. Assuming that a_1 and a_2 are independent, we arrive at the following expected utility.

$$EU[a_1(r_1), a_2(R-r_2)] = p_1p_2(u_1+u_2)+(1-p_1)p_2u_2 + p_1(1-p_2)u_1 - (1-p_1)(1-p_2)N \quad (1)$$

$$= p_1u_1+p_2u_2-N(1-p_1)(1-p_2) \quad (2)$$

Since N is assumed to be much larger than u_1 and u_2, the N term dominate the expected utility. Hence, maximizing the expected utility requires minimizing this term, which in turn involves reducing the probability of failing to complete the computation. Suppose we know that the resource requirement of a_1 is larger than that of a_2, but we do not have any further quantitative information (e.g., bounds) about their resource needs, we have the following qualitative probabilistic knowledge:

$$P(C_1|RA_1 = R) > P(C_2|RA_2 = R) \quad (3)$$

where C_i and RA_i represents "the step a_i completes" and "the amount of resource allocated to a_i" respectively.

Since $P(CS_i|RA_i=r_i)=P(C_i|RA_i=r_i)*P(S_i|C_i)$, we can derive the following qualitative formula from Equation (3) and the assumption $P(S_1|C_1) \geq P(S_2|C_2)$:

$$P(CS_1|RA_1 = R) > P(CS_2|RA_2 = R) \quad (4)$$

Thus, lacking more accurate information about resource needs of each computation steps, we can only maximize the expected utility simply by allocating all available resources to the task with the smallest resource requirement. Thus the reactive technique suggests that to ensure a feasible solution we must first execute the task with the smallest resource requirements. Our discussion can be easily generalized to the following resource allocation decision.

Decision 1 Suppose a_1, a_2, ..., a_n are atomic computation steps for solving a problem that has a deadline R. If all the following conditions are satisfied, then the optimal decision is to allocate all resources to a_1.

1. The utility of successful computations are additive:

$$U[CS_s, CS_j] = U[CS_s]+U[CS_j] \quad (5)$$

where S denotes a set of steps, and CS_s denotes successfully completing all steps in S.

2. The utility of not successfully completing any steps is a negative number (-N) that is much larger than the utility of completing all steps (i.e., $N \gg \Sigma U[CS_i]$).

3. $P(C_1|RA_1 = R) > P(C_i|RA_i = R)$ for all i, $1<i<= n$.

4. $P(S_1|C_1) >= P(S_i|C_i)$ for all i, $1<i<=n$.

The decision above only partially justifies reactive imprecision, for it only applies when there is a large negative utility associated with the worst possible outcomes. If the computation has progressed such that we have obtained some minimally acceptable solution, this assumption no longer holds. The problem of resource allocation in this case becomes the following:

Given that some computation steps have generated a result of minimally acceptable quality u_1, how to allocate remaining computing time available R′ among steps $a_1, a_2, \ldots, a_n$.

Rather than assuming a big negative utility for the worst outcome, it is more reasonable to assume a zero utility for not successfully completing any further computation steps. The expected utility of a resource allocation decision is

$$EU = \Sigma p_j u_j \qquad (6)$$

where p_j and u_j stands for $P(CS_j | RA_j = r_j)$ and $U[CS_j]$ respectively. Assuming that the utility of successfully completing each computation step is about the same, and that the resource needed to complete a, is less that resource needed to complete any other step. If we also assume that the likelihood for a_1 to successfully generate some solutions given enough resource is not less than that of other steps, we have the following inequality:

$$P(CS_i|RA_i=R') > P(CS_k|RA_k=R') \qquad (7)$$

Thus, allocating all resources to a_1 yields the maximal expected utility. This second decision underlying reactive imprecision is summarized below.

Decision 2 Suppose that we have obtained a solution of utility u, a_i, a_{i+1}, ..., a_n are remaining atomic computation steps for solving a problem that has a remaining deadline R′. If all the following conditions are satisfied, then the optimal decision is to allocate all resources to a_i.

1. The utility of successful computations are additive:

$$U[CS_s, CS_j] = U[CS_s]+U[CS_j] \qquad (8)$$

where S denotes a set of steps, and CS_s denotes successfully completing all steps in S.

2. The utility of all computation steps are about the same, i.e.,

$$U[CS_j] = U[CS_k] \qquad (9)$$

3. $P(C_i|RA_i = R) > P(C_k|RA_k = R)$ for all k, $i<k<=n$.

4. $P(S_i|C_i) >= P(S_k|C_k)$ for all k, $i<k<=n$.

As can be seen from the discussion above, allocating resources to the computation step that requires minimal resources can be easily justified in the situation where not producing any results

is catastrophic. However, continuing allocating resources to computation steps in a predetermined order after a minimally acceptable solution has been generated requires a stronger assumption about the relative utility of the steps (i.e., they are about the same).

2. The Foundation of Predictive Imprecision

To investigate the theoretical foundation of predictive imprecision, we consider the following problem:

Suppose that a problem needs to be solved within deadline R. A program to solve the problem consists of n steps a_1, a_2, ..., a_n, each of which requires t_1, t_2, ..., t_n to complete respectively. If the resource available is less than the total resource required how should the resource be allocated among the computation steps?

Since we have perfect information about the resource requirements, we know that a computation will complete for certain if it is given enough resources:

$$P(C_i|RA_i = r) = \begin{cases} 1 \text{ if } r \geq t_i, \\ 0 \text{ if } r < t_i \end{cases} \quad (10)$$

Therefore, a reasonable allocation of resource to step a_1 is either t_i or 0. In other words, the resource allocation problem in this case is equivalent to a binary optimization problem that determines what steps to skip. The expected utility of skipping a collection of steps S is

$$EU = \Sigma P(S_i|C_i)u_i. \quad (11)$$

If we further assume that the additional utility introduced by successfully completing each step is about the same, an optimal decision, then, is to remove steps with maximal resource needs until the resource requirement of the remaining steps can be satisfied by the resource available.

Decision 3 Suppose that a problem needs to be solved within deadline R. A program to solve the problem consists of n steps a_1, a_2, ..., a_n, each of which requires t_1, t_2, ..., t_n to complete respectively. If the resource available is less than the total resource required (i.e., $R < \Sigma t_i$) and the following conditions are satisfied

1. The utility of successful computations are additive:

$$U[CS_s, CS_j] = U[CS_s] + U[CS_j] \quad (12)$$

where S denotes a set of steps, and CSs denotes successfully completing all steps in S.

2. The utility of all computation steps are about the same, i.e.,

$$U[CS_j] \cong U[CS_k] \quad (13)$$

3. $t_1 <= t_2 \ldots <= t_n$

4. $P(S_i|C_i) \cong P(S_k|C_k)$ for all k, $i < k <= n$.

find the largest number k such that

$$\Sigma t_i <= R \quad (14)$$

and skip steps a_k through a_n.

We have discussed the decision-theoretic foundations of several resource allocation strategies in imprecise computation. Using assumptions about the independence of computation steps,

additive nature of their utility and qualitative information about relative utility and relative probabilities, we arrive at compiled efficient decisions that require very little resources themselves, and yet can respond effectively to dynamic availability of resources in a real-time environment.

Ⅲ. Summary

We have discussed a decision-theoretic foundation of imprecise computation techniques. Based on the theoretical framework, we have shown that both reactive imprecision and predictive imprecision can be viewed as partially compiled resource allocation decisions. Furthermore, we have shown that the underlying assumptions of these technique can be expressed as qualitative formula regarding resource needs and the expected utility of computation steps. Presenting imprecise computation techniques as qualitative decision-theoretic approaches to resource-bounded computing not only improves our understanding about the techniques but will also enable researchers to develop hybrid resource allocation techniques that can explore the trade-offs between the optimality and the efficiency of real-time computing by combining qualitative and quantitative decision-theoretic resource allocation strategies.

References

1. K.J. Lin, S.Natarajan, and J.W.S.Liu. "Imprecise results: Utilizing partial computations in real-time systems.", In Proceedings of the 8th Real-Time Systems Symposium, pages 210-217, San Jose, CA, 1987.
2. Swaminathan Natarajan. "FLEX: Towards Flexible Real-Time Programs", Computer Languages, vol. 16, no.1, pp. 65-79, 1991.
3. Thomas Dean. "An analysis of time-dependent planning", In Proceedings of AAAI-88, pages 49-54, 1988.
4. E.J. Horowitz. "Reasoning under varying and uncertain resource constraints". In Proc, National Conf. on Artificial Intelligence, pages 111-116, Minneapolis, MN, August 1988.
5. S. J. Russell and E.H. Wefald. "Principles of metareasoning.", In R.J. Brachman, H.J.Levesque, and R.Reiter, editors, Proceedings of the First Inter. Conf. on Principles of Knowledge Representation and Reasoning, Toronto, May 1989. Morgan kauffmann.