실시간 시스템에서 검사점 작성을 하는 태스크의 최악 수행시간 분석

김상수[O] 홍지만 조유근
서울대학교 전기 컴퓨터 공학부
{sskim[O], gman, cho}@ssrnet.snu.ac.kr

Analysis of Worst Case Execution Time of Tasks with Checkpointing in Real-Time Systems

Sangsu Kim[0], Jiman Hong, Yookun Cho
School of Computer Engineering, Seoul National University

요 약

검사점 작성을 이용하는 실시간 태스크의 스케줄링 가능성을 알기 위한 선행 조건으로 최악 수행시간을 분석하고 이를 최소로 하는 효율적인 검사점 작성의 위치를 결정하는 방법을 제시한다. 여기서 사용하는 조건은 k 개의 연속적인 결함을 허용하고 태스크의 검사점 작성 비용이 고정적인 경우와 가변적인 경우를 가정한다. 이러한 각 조건에서 최악 수행 시간을 최소로 하는 검사점 작성 알고리즘을 제시한다.

1. 서 론

검사점 작성은 프로그램이나 시스템의 수행 상태 정보를 저장해서 시스템의 심각한 오류나 결함이 발생하더라도 오류가 나기 전의 상태로 복구할 수 있도록 해주는 기술이다. 이를 통해 오랜 시간 동안 수행을 해야 하는 작업이 도중에 시스템의 결함이 발생하더라도 처음부터 다시 시작할 필요 없이 임의의 이전 검사점 위치에서 수행을 재개할 수가 있다. [3]

실시간 시스템에서 이런 검사점 작성 기법을 이용하면 태스크의 수행 시간을 단축할 수가 있고 결함 허용도를 높힐 수가 있다.[1] 실시간 시스템에서 태스크를 주어진 데드라인의 조건을 만족하면서 스케줄링을 해야 하는 경우에 실시간 스케줄링을 위해서는 최악 수행 시간 (worst case execution time: WCET)의 정보가 필수적이다. 그러나 검사점 작성을 어떻게 언제 하느냐에 따라 그리고 결함 발생 여부에 따라서도 실제 수행 시간이 달라지므로 최악 수행시간을 주어진 조건에 따라 계산해야 한다.

이 논문에서는 이러한 검사점 작성 태스크의 최악 수행시간을 구하고 이를 최소로 하는 검사점 작성 방법에 대해서 논한다.

2. 조건 및 가정

이 논문에서 가정하는 실시간 태스크는 단일 또는 다중 태스크로 주어지고 단일 처리기 모델을 가정한다. 그리고 데드라인과 태스크 도착 시간 등은 고려하지 않는다.

검사점 작성에서 각 검사점 작성 및 복구에 필요한 추가적인 수행 시간은 검사점 비용과 복구시의 복구 비용으로 나타내고 이들 모두 고정적인 값 또는 가변적인 값을 가질 수 있다. 일반적으로 검사점의 비용은 검사점 정보의 크기에 대체로 비례하고 그 크기는 메모리 정보에 가장 큰 영향을 받는다. 그리고 복구 비용은 해당 검사점의 크기에 대체로 비례하는 경향이 있다.

결함 모델은 최대 k 개의 결함이 발생함을 가정한다.

이 논문에서는 다음의 기호를 사용한다.

T	태스크의 순 수행시간
T_w	태스크의 최악 수행 시간
c	검사점 작성 비용
r	복구 비용
k	최대 결함 발생 회 수
n	검사점 작성 회수

3. 검사점 작성 태스크의 최악 수행시간

여기서 제일 먼저 단일 태스크, 고정 검사점 비용 모델에서 최악 수행 시간을 구하고 다중 태스크의 경우 각 태스크마다 검사점 비용이 다르고 각 태스크 수행 동안에는 그 비용이 일정한 모델을 고려한다.

3.1 고정 검사점 비용 모델

태스크는 하나로 주어지고 검사점 작성 및 복구 비용은 상수인 모델이다. 여기서 가장 기본적인 등간격 검사점에 관한 다음과 같은 정리를 얻을 수 있다.

정리 1. 등간격 검사점

등간격 (equidistance)으로 검사점을 작성시 최악 수행 시간은 다음과 같이 주어지고

$$Tw = T + nc + k\left(r + \frac{T}{n}\right)$$

이를 최소로 하는 최적의 검사점 작성 회수 n은 다음과 같다.

$$\left\lfloor \sqrt{\frac{kT}{c}} \right\rfloor \quad if \quad T < \frac{cn(n+1)}{k},$$

$$\left\lceil \sqrt{\frac{kT}{c}} \right\rceil \quad if \quad T \geq \frac{cn(n+1)}{k}$$

증명: 최악 수행시간 Tw는 태스크의 순 수행시간 T, 전체 검사점 작성 비용의 합 nc, 그리고 결함 발생시 최대 k번의 복구 비용의 합 $k(r + T/n)$으로 이루어진다. 최악의 경우 발생은 결함이 어느 구간의 검사점 작성직전에 발생하는 경우로 한 번의 결함으로 최대 한 구간만큼의 시간과 복구 비용이 필요하게 된다. 최악 수행시간의 식을 n에 관해 미분을 하게 되면 다음과 같은 식을 얻을 수 있다.

$$\frac{\partial Tw}{\partial n} = c - \frac{kT}{n^2}$$

또한 $\partial^2 Tw/\partial n^2 > 0$ 이므로 $\partial Tw/\partial n = 0$ 이 되는 $n = \sqrt{kT/c}$ 지점에서 최소값을 가진다. 따라서 n이 $\lfloor \sqrt{kT/c} \rfloor$ 또는 $\lceil \sqrt{kT/c} \rceil$에서 최소값을 가지게 되는데 n, n+1 두 경우의 Tw 값을 비교해서 부등식을 풀면

$$T + nc + k\left(r + \frac{T}{n}\right) < T + (n+1)c + k\left(r + \frac{T}{(n+1)}\right).$$

$$\Rightarrow T < \frac{cn(n+1)}{k}$$

이므로 위 정리의 결과를 얻을 수 있다. ■

검사점 작성에 걸리는 시간 c가 언제나 동일한 경우 이와 같은 등간격 검사점 작성 방법이 최적의 WCET를 가짐은 다음과 같이 쉽게 보일 수가 있다.

일반적으로 n개의 검사점 작성을 할 때 각 구간의 시간을 순서대로 T_i 라고 하면 다음과 같이 WCET가 주어진다.

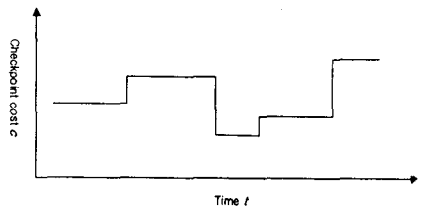
$$Tw = \sum_{i=1}^{n} T_i + nc + k\max(r + T_i)$$

$$= T + nc + kr + k\max(T_i)$$

여기서 $\sum_{i=1}^{n} T_i = T$ 이고 $\max(T_i) \geq T/n$의 관계가 성립하므로 등간격으로 구간을 나누는 경우에 최소값을 가짐을 알 수 있다.

3.2 가변 검사점 비용 모델

보다 일반적인 경우로 검사점 비용이 검사점 작성 시기에 따라 가변적인 경우를 생각해 보자. 검사점 비용의 변화는 주로 실행중인 태스크의 메모리 사용량이 변화와 비슷한 유형으로 증감하는데 대체로 일정 구간동안 그 크기가 유지되고 메모리 동적 할당과 해제에 따라 증감이 계단 형으로 일어나서 다음 그림과 같은 유형의 변화를 보이는 것이 일반적이다.[2]



그래서 여기서는 이러한 계단형의 검사점 비용 변화 모델을 가정하고 같은 검사점 비용을 가지는 구간을 각각 순서대로 T_i 라고 하고 전체 m개의 구간이있다고 하자. 그리고 각 구간에서는 등간격으로 검사점 작성을 n_i 번 한다. 그러면 WCET은 다음의 정리와 같이 주어진다.

정리 2. 가변 검사점 모델

가변 검사점 모델에서 최악 수행시간은 다음과 같다.

$$Tw = \sum_{i} (T_i + n_i c_i) + k\max\left(r_i + \frac{Ti}{n_i}\right)$$

그리고 다음과 같은조건을 만족할 때 이 값은 최소가 된다.

$$r_1 + \frac{T_1}{n_1} = r_2 + \frac{T_2}{n_2} = \ldots = r_m + \frac{T_m}{n_m}$$

증명: 최악 수행 시간은 각 구간의 검사점 작성 비용과 결함 발생시 일어날 수 있는 최악의 복구 비용 합하면 위 식이 나온다. 이 식에서 보듯이 WCET 는 복구 비용과 롤백하는 구간의 시간의 합이 제일 큰 구역에서 결함이 발생 했을 경우에 발생한다. 따라서 WCET를 최소로 하려면 복구 비용의 최대값을 줄여야 한다.

여기서 Tw 값을 최소로 하는 n_i 값을 구하려면 우선 각 n_i 값이 서로 독립적인 변수이므로 각각에 대해 편미분을 해서 극소값을 구하면 된다. 일반성을 잃지 않고 $r_1 + \frac{T_1}{n_1} \le r_2 + \frac{T_2}{n_2} \le \ldots \le r_m + \frac{T_m}{n_m}$ 이라고 가정하자, Tw은 마지막 m번째를 제외한 나머지 $n_i \quad (1 \le i < m)$에 대해 $k \max\left(r_i + \frac{Ti}{n_i} \right)$ 부분이 상수가 되므로 오직 $n_i c_i$ 만 최소로 하면 된다.

따라서 이 경우 n_i를 $r_1 + \frac{T_1}{n_1} \le r_2 + \frac{T_2}{n_2} \le \ldots \le r_m + \frac{T_m}{n_m}$ 조건에만 맞는 최소한의 값으로 하면 된다.

$i = m$ 인 경우는 정리 1에 의해 $n_m = \left\lfloor \sqrt{kT_m / c_m} \right\rfloor$ 일 때 극소값을 가지게 된다.

결론적으로 다음 조건을 만족할 때 Tw 값은 최소가 된다.

$$r_1 + \frac{T_1}{n_1} = r_2 + \frac{T_2}{n_2} = \ldots = r_m + \frac{T_m}{n_m}$$

위 정리는 다시 말해서 최악 수행시간을 최소로 하는 경우는 복구시의 비용이 가장 큰 구간을 기준으로 나머지 구간들이 이 값을 넘지 않는 범위에서 최소의 검사점을 작성하면 된다는 것을 의미한다.

만약 T_m 이 너무 짧아서 n_m의 값이 0이 되는 경우도 발생할 수 있는데 여기서는 검사점 작성 비용이 일정한 구간이 충분히 길어서 그러한 경우가 없다고 가정한다.

3.3 적응적인 검사점 스케줄링

가변적인 검사점 모델에서 중요한 결정 요소가 롤백 및 복구시의 비용이 가장 높은 구간에서의 검사점 작성 인터벌이다. 따라서 만약 태스크가 어느 한 검사점을 완전히 작성하면 그 태스크는 더 이상 그 검사점 이전으로는 복구할 필요가 전혀 없어지게 되므로 그 검사점 이전의 구간에서 검사점 비용의 최대 구간이 있었다면 이후의 구간에 대해서 다시 최대 복구 비용 구간을 잡고 검사점 인터벌을 재계산하면 된다. 예를 들어 복구 비용이 이렇게 매 검사점 작성 시기마다 다시 검사점 인터벌을 적응적으로 계산하는 것을 적응적인 검사점 스케줄링이라고 한다.

이뿐만 아니라 만약 이전에 결함이 발생되어 앞으로 남은 최대 결함 회수에도 변화가 생기면 이 또한 적응적으로 검사점 작성 인터벌을 계산해야 한다.

다음은 이를 종합적으로 반영한 검사점 작성 인터벌을 결정하는 알고리즘이다. 기본적인 아이디어는 계산의 효율을 위해 매 검사점 작성시마다 구간 계산을 하지 않고 검사점 구간을 태스크의 마지막 부분부터 처음으로 거슬러 올라가면서 최대의 복구 구간이 생기는 부분을 유지해 가면서 최대한 이 값을 넘지 않는 범위에서 검사점 구간을 최대로 하는 값으로 구간을 정하는 것이다.

즉 m부터 i까지 거슬러 올라가면서 T_m부터 T_i까지 정리 2의 조건을 만족하도록 구간을 계산하는 것이다. 그리고 알고리즘에서는 검사점 작성 회수가 아닌 검사점 작성 인터벌을 구한다. 즉 interval[i]는 T_i 구간에서의 검사점 작성 인터벌이 되는 것이다.

```
compute_checkpoint_interval:
  max_recover_cost := 0;
  max_rollback_cost := 0;
  for i = m..1
    if recover_cost [i] < max_recover_cost
    then
        interval[i] := max_rollback_cost – recover_cost[i]
    else
        max_recover_cost := recover_cost[i]
        interval[i] := sqrt(c_i T_i / k)
        max_rollback_cost := recover_cost[i] + interval[i]
```

4. 결론

이 논문에서 실시간 시스템에서 검사점 작성을 이용하여 결함 허용성과 최악 수행 시간을 최소로 효율적인 검사점 작성의 위치를 결정하는 방법을 제시하였다. 검사점 비용이 일정한 경우에는 등간격 검사점 작성 인터벌 방법으로 최악 수행 시간을 최소로 할 수 있고 가변 검사점 모델에서는 복구 비용이 제일 큰 구간이 나머지 이후의 구간에 결정적인 요소로 검사점 작성 인터벌을 결정함을 알 수 있었다. 앞으로 검사점 작성과 복구 비용의 변화가 단순한 계단형이 아닌 좀더 복잡하게 변화하는 일반적인 경우에 최적의 검사점 작성 알고리즘의 연구가 요구된다.

참고 문헌

[1] R. Bettati, N. S. Bowen, Jen-Yao Chung , *On-Line Scheduling for Checkpointing Imprecise Computation,* Proceedings of Euromicro 93 Workshop on Real-Time ystems, June 1993

[2] J. Hong, S. Kim, Y.Cho, "On the choice of checkpointing interval using memory profile and adaptive time series analysis", *In Proceeding of IEEE Pacific Rim Symposium on Dependable Computing*, Dec. 2001

[3] D. Prdhan, *Fault-Tolerant Computing System Design*, John Wiley, 1996