ATM을 위한 프로세스 스케줄링에 관한 연구

유석대[0] 박지연 조기환 이문근
전 북 대 학 교 컴 퓨 터 과 학 과
{sdyu, jypark, ghcho, mklee}@cs.chonbuk.ac.kr

A Study on Process Scheduling of ATM

Suk-Dea Yu[0] Ji-Yeon Park Gi-Hwan Cho Moon-Kun Lee
Dept. of Computer Science, Chonbuk National University

요 약

ATM을 이용한 명세에 대한 스케줄링 가능성 분석은 명세된 실시간 시스템의 시간적 제약이 만족되는지를 검증하기 위해 필요한 작업이다. 본 논문에서는 ATM으로 작성된 명세에 스케줄링에 필요한 요소를 추가하여 머신의 형태나 자원의 형태에 따른 다양한 스케줄링 알고리즘을 적용함으로써 ATM 명세가 스케줄링 가능하게 하는 것을 보인다.

1. 서론

실시간 시스템은 병렬성, 통신, 동기/비동기, 입·출력과 같은 다양한 속성과 분산성, 실패 복구(*fault tolerance*) 등과 같은 환경적 성질과 관련되어 있어 비실시간 시스템에 비해 높은 복잡도를 가지며 그 규모 또한 방대하다. 규모가 크고 복잡한 실시간 시스템을 적은 비용과 노력을 들여 안전성이 보장된 개발을 하기위해 다양한 패러다임의 정형적 개발 방법이 사용되고 있다.

ATM(*Abstract Timed Machines*)[1,2,3]은 방대하고 복잡한 실시간 시스템을 명세하기위해 개발한 정형 기법으로 병렬적으로 실행되는 시스템 요소와 통신, 우선 순위, 시간, 자원 사용과 같은 다양한 속성의 명세가 가능하다. 그러나, 실시간 시스템 개발 과정에서 명세의 정확성을 지원하고 시스템의 안정성을 보장하기 위해서는 명세한 시스템에 대한 검증, 모의 실험 등이 구현 이전 단계에서 수행되어야 한다. 이를 위해 필요한 속성 중의 하나가 스케줄링 개념이다.

본 논문에서는 ATM을 이용하여 명세한 시스템이 명세 단계에서 스케줄링을 지원할 수 있도록 스케줄링에 필요한 명세 속성을 정의하며, 정의된 속성이 추가된 ATM을 이용하여 명세한 시스템이 여러 스케줄링 방법을 적용하였을 경우 스케줄링 가능함을 보인다. 즉, 명세 단계에서의 시스템의 스케줄링 가능성을 분석하여 시스템의 안정성과 효율성을 높일 수 있음을 기술한다. 스케줄링에 대한 예를 보여 ATM을 이용한 명세의 스케줄링 가능성을 보인다.

2. 관련연구

실시간 시스템의 스케줄링 기법은 프로세스와 자원 그리고 우선 순위와 관련된 처리 방법에 따라 다양한 기법과 알고리즘이 제시되었다[4,5,6,7].

[4]에서는 ACSR-VP(*Algebra of Communicating Shared Resources with Value Passing*)를 이용하여 명세된 실시간 시스템의 스케줄링 가능성을 등가(*equivalence*) 검사에 근거하여 분석하였고, 이로써 명세도구인 ACSR-VP가 스케줄링 가능함을 보였다.

[7]은 시간 제약에 따른 효과적인 스케줄링 알고리즘을 적용하는 방법론을 기술하며, 스케줄링에 필요한 요구사항과 유형별로 효과적인 스케줄링 방법을 제안하고 있다.

스케줄링은 프로세스 선점 가능 유무와 다른 프로세서들로 이동 가능 유무에 따라 여러 가지 제약이 존재하며 전반적인 알고리즘에 많은 영향을 준다. 따라서 명세하려는 시스템의 작업들과 자원들의 속성, 명세 방법 등에 따라 적절한 알고리즘 선택은 중요하다.

본 논문의 구성은 3장에서는 ATM에 대하여 기술하고 4장에서는 ATM을 스케줄링 하기 위해 필요한 요구사항과 효과적인 알고리즘의 적용과 그 예를 보인다. 마지막으로 5장에서 결론과 향후 연구에 대하여 기술한다.

3. ATM

3.1 ATM 구성 요소

ATM은 크게 머신 단위로 시스템을 명세하며 머신은 모드와 전이, 시작점과 종료점, 다른 머신과의 상호 작용을 나타내는 포트를 가진다. 각 구성 요소는 해당 레이블을 갖는다[3].

머신은 태스크나 프로시져와 같은 독립적 프로그램 블록 단위를 나타내는 명세 단위로 활성화 되었을 때 내부의 흐름을 스스로 제어할 수 있다.

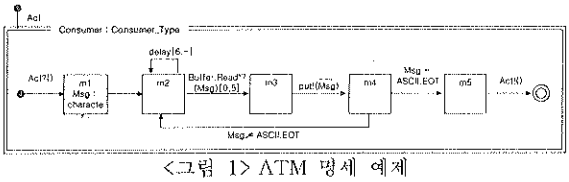
ATM의 모드는 일반적인 계산을 위한 계산 모드, 추상화를 위한 추상화 모드, 예외 처리나 주기적 작업 등을 명세하기 위한 주제 모드가 있다. 각 모드는 이름, 아이디, 타입, 시간 제약 등 자신을 대표할 수 있는 속성을 갖는다.

본 연구는 한국과학재단 특정기초연구(과제번호 1999-2-303-003-3) 지원으로 수행되었음

전이는 모드에서 모드로의 제어의 변환을 나타내며, 이벤트, 조건, 시간 제약, 통신에 관련한 속성을 표현한다.

포트는 머신 간의 통신을 나타내는 것으로 포트를 통하여 각 머신은 상호 작용을 한다. 포트는 머신의 활성/비활성을 위한 활성화(*activation*), 엔트리(*entry*), 대체(*substitute*) 포트가 있다.

<그림 1>은 ATM 구성 요소를 사용한 명세 예이다. 머신은 모드와 전이로 구성되어 있으며 시작점으로부터 시작되어 각 제어의 변환이 모드와 전이를 거쳐 명세되어 종료점에 이른다. 머신은 활성화를 위한 활성화 포트를 가지고 있다.



<그림 1> ATM 명세 예제

3.2 주기성과 비주기성

ATM은 실시간 시스템이 가진 속성 중 시간 개념을 가진 주기 작업과 비주기 작업을 명세하기 위해 다양한 형태의 시간 명세 방법을 정의하고 있다[1]. 본 절에서는 ATM의 스케줄링과 관련한 주기적 시간 명세와 비주기적 시간 명세에 대해 대략적으로 기술한다.

3.2.1 비주기적 시간 명세

시간 제약을 가진 비주기적 작업은 크게 두 종류의 시간 제약을 가진다. 상한(*lower bound*)과 하한(*upper bound*) 시간 값 내에 발생하거나 특정 시간 동안 실행이 지속되어야 한다. 또는 이 두 조건이 복합된 시간 제약을 가진다. [1, 2, 3]에서 정의한 ATM은 모드와 전이에 이러한 제약이 표현된다. <표 1>은 비주기적 시간제약에 대한 시간 명세 문법이다.

구 분	*syntax*
between deadline	*[lower bound time, upper bound time]*
duration execution	*[execution time]*
composition	*[lower bound time, execution time, upper bound time]*

<표 1> 비주기 시간 제약 문법

3.2.2 주기적 시간 명세

ATM 명세 기법은 실시간 시스템이 가진 주기적 작업을 명세할 수 있도록 하기 위해 모드와 전이가 주기적 시간 제약을 가질 수 있도록 정의하였다. 주기적 시간 작업의 경우 비주기적 시간 제약에 추가된 *delay*, 한 주기 시간, 한 주기 작업이 완료되는 *deadline* 시간을 정의한다. <표 2>는 주기적 시간 제약에 대한 문법이다. 전체 주기 작업은 특정 시간 내에 시작하여야 하며, 일단 작업이 시작되면, *delay* 이후 주기 시간동안 실행 시간(*execution*)만큼 실행하며, 한 주기의 실행은 *deadline* 내에 끝나야 한다. 주기적 작업에 대한 시간 제약은 반복 회수 *n*을 명세하여 주기적 작업의 실행에 대한 명확한 명세를 제공한다.

구 분	*syntax*
periodic execution	*[lower bound, (delay, period, execution, deadline)$^{[n]}$, upper bound]*

<표 2> 주기적 시간제약 문법

3.3 우선 순위

ATM은 각 머신에 우선 순위를 명확히 명세하여 자원 사용이나 실행의 선후에 대한 스케줄링을 돕는다.

3장에서는 ATM에 대한 구성 요소와 스케줄링에 관련 있는 ATM 정의를 살펴보았다. 다음 장에서는 ATM의 스케줄링성을 돕기 위한 추가적 요구 사항과 이를 통한 스케줄링 과정을 기술한다.

4. ATM에서 스케줄링

4.1 스케줄링에 필요한 요구사항

시스템의 스케줄링은 스케줄링을 위한 다양한 요소를 필요로 한다. [7]에서는 이러한 조건을 분석 기술하고 있으며 본 논문은 이에 근거하여 스케줄링 가능성을 분석한다.

스케줄링 시 필요한 모드들의 선행 관계와 머신이 가지는 우선 순위의 명세이다. 머신의 선행 관계는 시간이나 우선 순위에 의해 변경 될 수 없는 값이므로 정적으로 고정되어있다고 할 수 있다. ATM에서 스케줄링을 위해 필요로 하는 값들은 머신의 속성과 시간 속성 *release time* , *execution time*, *deadline*이다. 또 각 머신에는 우선 순위가 설정할 수 있어야 한다. 기존 ATM 정의는 모드와 전이에만 *release time*, *execution time*, *deadline*의 시간에 대한 속성을 정의하고 있다. 스케줄링을 지원하기 위해 ATM 머신에 또한 이러한 시간 정의가 추가 되어야 한다. 우선 순위와 머신 간의 선후 관계는 기존의 ATM에 정의 되어 있다.

ATM은 때문에 시간에 대한 다양한 표현력을 가지고 있다. ATM 명세를 스케줄링 가능하게 하기 위해서는 다양한 시간의 속성에 맞는 알고리즘들이 적절히 적용되어져야 한다.

4.2 스케줄링 가능한 ATM

ATM의 머신을 스케줄링하기 위해 필요한 시간 속성이 추가된 머신 레이블은 다음과 같다.

(<caller_name>)(<container_name::>)

<machine_name>(<parameter_list>)([time_list])(<:machine_type>)

(<# priority_number>)(<machine_properties_list>)

<표 3> ATM 머신 레이블

*<caller_name>*은 머신들의 선후관계를 표현하고 있다. 스케줄링의 방법론 선택에 중요한 요소인 선후관계를 머신간의 호출을 이용하여 표현할 수 있다.

*[time_list]*는 머신의 시간 속성을 정의하는 리스트로서 *release time*, *execution time*, *deadline*을 포함하고 있다. 시간의 신택스는 시간 제약성이 없을 때는 생략가능하고, *execution time*의 경우 범위를 가지는 값도 가능하다. 호출 관계에 있는 머신의 경우 시간의 속성을 상속 받아 특별히 기입하지 않아도 선행 머신에 의해서 시간적 속성이 주어진다.

*<#priority_number>*는 우선 순위를 명세하는 요소로 함수에 의해 정적으로 정의 되는 $\alpha(= f(x)$: x는 우선 순위 변화 가중치, f는 우선 순위 함수)가 할당된다. 적용하려고 하는 머신의 형태나 스케줄링 방법론에 따라 항등함수, 단조 증가함수 또는 식에 의한 함수의 값을 가진다.

*<machine_properties_list>*는 머신의 속성을 정의하는 리스트로서 *preemptable*, *migratable*을 포함하고 있다. 정의 하지 않았을 때는 기본값으로 정의된다. *preemtable*에 *T*가 할당되면 그 머신은 선점가능하고, *F*이 할당되면 선점할 수 없다. *migratable*에 *M*이 설정되면 프로세서간 머신의 이동이 가능하고, *F*이 할당되면 이동이 불가능하다.

4.3 머신 정의에 의한 스케줄링 적용

4.3.1 머신의 주기에 따른 스케줄링

주기적 작업 위주의 스케줄링은 비교적 간단하며 예측이 가능하지만 시간에 의해서만 스케줄링이 이루어지기 때문에 작업들이 *deadline*을 넘어서는 경우가 많이 발생한다. ATM 명세의 경우에는 *periodic execution*과 *between-deadline. duration execution*을 이용하여 주기적 및 비주기적 작업을 명세 할 수 있다. 머신이 가지는 시간제약은 선택된 알고리즘에서 머신의 스케줄링을 위한 우선 순위 설정에 기준 값으로 사용된다.

4.3.2 우선 순위에 의한 스케줄링

ATM 명세에서 우선 순위는 함수 값을 가지고서 할당이 된다. α는 정적 우선 순위를 가진 경우에는 항등식 $f_i(identity)$나 단조 증가 $f_m(monotonic)$을 가지고 정의되며, 동적 우선 순위를 가지는 경우에는 함수식 f_x를 이용하여 정의된다. ATM은 적용되는 함수식에 따라 *FIFO, shortest-processing-time-first, earliest-deadline-first*[EDF] 그리고 *rate-monotonic*의 알고리즘들이 선택적으로 사용한다.

4.3.2 자원의 종류에 따른 스케줄링

ATM을 이용한 명세가 특정 머신을 선점 가능하게 했을 때와 그렇지 못하게 했을 때의 스케줄링은 크게 달라지게 된다. 일반적으로 CPU와 같이 선점 가능한 자원과 프린터와 같이 선점이 불필요한 자원이 있다. ATM에서 명세한 선점 가능한 자원 머신은 *priority-ceiling protocol*[9]에 의해 자원에 대한 스케줄링을 수행하여 우선 순위 도치[9]와 같은 우선 순위 도치 현상을 방지한다. 선점 불가능한 대기 큐에서는 우선 순위에 의하여 스케줄링만 허용하여 해당 머신의 특성을 보존할 수 있다.

4.4 ATM 명세를 이용한 스케줄링 예제

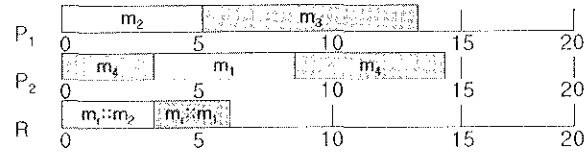
m_1, m_2, m_3, m_4, m_r(자원머신)의 머신이 있고 레이블이 다음과 같다고 할 때 주어진 레이블이 스케줄링 가능한 경우와 가능하지 않은 경우를 살펴 보겠다.

머신	레이블
m_1	- [0, 10, 5], α_1, <P,N>
m_2	- [0, 8, 5], α_2, <P,N>
m_3	- [4, 15, 8], α_3, <P,N>
m_4	- [0, 20, 10], α_4, <P,N> and <N,N>
m_r	- [, , 3], α_r, <N,N>

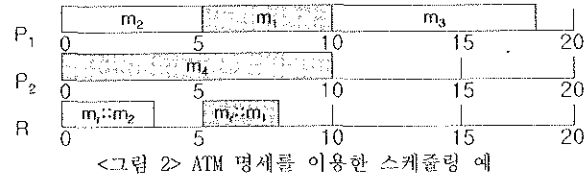
EDF알고리즘 사용
$\alpha_i = d_i$ - Current_time;
m_1과 m_2가 m_r사용

<표 4> ATM 머신 레이블 예제

m_4가 <P,N>인 경우



m_4가 <N,N>인 경우



<그림 2> ATM 명세를 이용한 스케줄링 예

다음 <표 4>의 ATM 머신 명세에서 머신 m_4는 선점 가능할 때와 불가능 한 경우를 가진다. <그림 2>는 두 경우에 대한 알고리즘에 의한 스케줄링 결과이다. 머신 m_4가 선점 불가능하게 될 때는 머신 m_3이 *deadline*을 넘게 되어 시간제약을 만족하지 못한다.

5. 결론 및 향후 연구과제

본 논문에서는 시간 제약에 대한 다양한 명세 방법을 지원하는 ATM 명세를 이용하여 스케줄링에 관한 요구사항과 이를 적용하는 방법론을 살펴보았다. 스케줄링 가능성은 머신에 시간 제약과 우선 순위, 우선 순위와 지원 함수 등의 속성들을 추가 정의 함으로써 지원된다.

또한 ATM으로 작성된 명세를 가지고 머신의 형태나 자원의 형태에 따른 다양한 스케줄링 알고리즘을 적용함으로써 ATM 명세가 스케줄링 가능하다는 것을 보였다.

향후 연구과제로서는 ATM으로 명세된 시스템을 시뮬레이션 할 수 있는 도구의 개발과 상황에 맞는 우선 순위 부여 함수의 모델화이다. 또 분산 모델에 적용하기 위한 추가적인 명세 정의가 필요하다.

참고 문헌

[1]. 노경주, 박지연, 이문근, "*추상 시간 기계를 기반으로 한 실시간 시스템을 위한 시간 명세와 분석*", 정보과학회 학술대회논문지(A), 제 27권 1호, 2000년, pp. 576-578

[2]. 박지연, 노경주, 이문근, "*추상 시간 기계를 사용한 실시간 시스템의 역명세 검증*", 정보과학회 학술대회논문지(A), 제 27권 1호, 2000년, pp. 489-491

[3] 박지연, 노경주, 이문근 "*순환공학을 위한 정형기법: 추상시간기계*" 한국정보과학회 소프트웨어공학연구회 학술발표논문지(KCSE-2000), 2000년 2월, pp.61-70

[4]. 최진영, "*ACSR-VP를 이용한 실시간 시스템의 스케줄링 기법에 대한 정형명세와 검증*", 정보과학회 논문지(A), 제 25 권 제 6 호, 1998년

[5]. J.Y.-T. Leung and J. Whitehead. "*On the Complexity of Fixed-Priority Scheduling of Periodic Real-Time Tasks*", Performance Evaluation, 2:237-250, 1982.

[6]. C.L. Lui and J.W. Layland. "*Scheduling Algorithm for Multiprogramming in a Hard Real-Time Environment*". Jounal of the ACM, 20(1):46-61, 1973.

[7]. J.W.S. Liu and R. Ha. "*Efficient Methods of Validating Timing Constraints*", the Real-Time System:199-223

[8]. J.P. Lehoczky and S.R. Thuel. "*Scheduling Periodic and Aperiodic Tasks Using the Slack Stealing Algorithm*", the Real-Time System:175-198, 출판사, 출판년도

[9]. L. Sha, R. Rajkumar and J.P. Lehoczky. "*Priority inheritance protocols: An approach to real-time synchronization*". IEEE Transactions on Computers, 39(9):1175-1185, Sep. 1990.