이동 분산 실시간 시스템을 분석하기 위한 DATM 이동 전이 규칙

이정희*0 박지연* 박주호* 이문근**
전북대학교 컴퓨터과학과*, 전북대학교 전자정보공학부**
{jehlee0, jypark, juhpark, mklee}@cs.chonbuk.ac.kr

Transition Rules of DATM movements for Analysis of Mobile Distributed Real-Time System

Jeong-Hee Lee*0 Ji-Yeon Park* Ju-Ho Park* Moon-Kun Lee**
*Dept. of Computer Science, Chonbuk National Univ.,
**Div. of Electronics and Information Engineering, Chonbuk National Univ.

요 약

이동 분산 실시간 시스템(*MDRTS :Mobile Distributed Real-Time System*)은 특정 시간 내에 분산 네트워크를 통한 정보 교환 및 요구된 동작을 실행한다. 그리고 경로, 시간 및 속력에 관한 제약사항에 따라 머신이 이동하는 시스템이다. 본 논문은 MDRTS에서 발생되는 머신의 이동을 분석하기 위해, PATM[1]에 위치 공간 개념을 추가하여 확장한 DATM(*Distributed Abstract Timed Machine*)을 정의한다. 머신의 이동과 이동을 제약하는 사항들, 경로, 시간 및 속력 등을 나타내는 표현법을 정의하며, 각 제약 사항에 따른 이동에 관한 규칙을 기술한다.

1. 서 론

최근의 항공, 위성, 국방 등의 사회 각 분야에서 사용되는 시스템은 머신의 이동에 관한 정보를 요구한다. 시스템에서 요구된 이동이 발생하지 못한 경우 시스템의 신뢰도는 급격히 감소하며 재산이나 인명의 심각한 위험을 초래한다.

DATM은 기존의 PATM[1]에서 명세할 수 없었던 MDRTS의 이동정보를 표현하기 위해 머신의 위치정보를 정의한다. DATM은 머신의 이동 경로를 이산 경로(*discrete path*)로 표현하며, 시스템 실행 시 발생하는 머신의 이동에 대한 분석을 하기 위해 머신의 이동을 정의한다. 머신의 이동에 관한 시간과 속력의 제약사항에 따라 이동의 규칙을 기술하며, 여러 머신의 이동에서 발생할 수 있는 머신의 충돌을 예측하는 방안과 방지하는 방법에 대해 살펴본다.

본 논문의 구성은 2절에서 머신의 이동의 관련 연구를 기술하며, 3절에서 DATM의 위치 정보와 머신의 이동 정보를 정의한다. 4절에서는 단일 머신과 여러 머신에서 이동에 관한 시간, 속력의 제약 사항에 따른 이동 규칙을 기술한다. 5절은 정의한 머신의 이동을 적용한 예제이며, 마지막으로 결론 및 향후연구이다.

2. 관련 연구

머신의 이동을 명세하고 분석하며, 머신이 이동시 머신 간의 충돌(*conflict*)을 방지하는 방법론들이 연구되고 있다.

머신의 이동 경로를 표현하기 위한 방법으로 연속 경로(*continuous path*)와 이산 경로(*discrete path*)가 있으며, 연속 경로는 물리 법칙에 의해 표현한다[3]. 머신들이 이동할 때 머신 간의 충돌은 예측 가능하며, 머신 간의 충돌 확률 계산법[4]가 연구되었다. 또한, 머신의 이동시 머신 간의 충돌을 방지하는 방법[3,4,5,6]도 연구되었다.

3. DATM의 이동

3.1 DATM의 위치와 시간

DATM은 이동 분산 실시간 시스템을 명세하기 위해 고안한 정형 기법으로, 실시간 시스템을 명세하기 위해 제안된 PATM에 분산 정보와 위치 정보를 추가하여 확장한 기법이다.

DATM의 위치 정보는 머신이 존재하는 지점이나 영역, 시간 등을 의미하며, 시간은 머신이 가진 시간제약을 의미한다.

정의 3.1 (DATM의 위치) 머신은 존재하는 위치는 특정지점(*point*)이거나 영역(*region*)이며 P_{name}으로 표기한다. 여기에서 *name*은 소문자이거나 대문자로, 소문자인 경우에는 지점을, 대문자인 경우에는 영역을 나타낸다. 지점은 절대 위치와 상대 위치로 구분하며 절대 위치는 '<,>'를, 상대 위치는 '(,)'를 사용하여 각각 2차원 좌표(<*x, y*>, (*x, y*)) 또는 3차원 좌표(<*x, y, z*>,(*x, y, z*))로 표현한다. □

3.2 머신의 이동

머신의 이동은 이동 조건에 따라 특정 위치에서 다른 위치로의 위치 변화를 의미한다. 이동 조건은 시간의 경과나 통신 이벤트, 현재 위치 값, 속력 등에 의해 표현된다. 머신이 이동할 때 이동 경로는 본 논문에서는 실제 시스템에 적용할 수 있는 이산 경로로 명세한다.

정의 3.2 (이동 경로) 머신의 이동 경로는 머신이 이동시 경유해야 하는 위치, 경유하지 말아야할 위치, 경유해도 되고 하지 않아도 될 위치들을 명세한 리스트{P_i, P_h,($\neg P_j$),($\neg P_j$), [P_i], [P_j]}로 정의한다. P_i와 P_h은 머신이 경유해야 하는 지점과 영역을, ($\neg P_j$)나 ($\neg P_j$)는 경유하지 말아야할 지점과 영역을, [P_i], [P_j]은 경유해도 되고 않아도 되는 선택적 지점과 영역을 의미한다. □

정의 3.3 (시간 제약) 머신의 이동에 관한 시간 제약은 [*lower_bound*, 〈*delay, execution, period*〉 ,*upper_bound*]로 표기한다. 머신의 이동은 *lower_bound*와 *upper_bound*사이에서 시작되고 완료되며, *delay*는 이동이 시작하기 전에 이동이 지연되어야 하는 시간이다. 머신의 이동 경로가 cycle 형태를 가질 경우, 머신은 *period* 시간마다 특정 경로로 반복적인 이동을 *execution*시간 동안 수행한다[2]. □

시간 제약은 절대 시간과 상대 시간으로 표현한다. 절대시간은 '@'을 사용하여 '@hour : minute : second'로 표기하며, 상대시간은 단위 시간으로 표현한다. 예를 들어 머신이 정각 12:00에 이동을 시작한다면 이동의 시작 시간을 절대시간 '@12:00:00'으로 표현한다.

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

정의 3.4 (속력 제약) 머신이 이동을 할 때 머신의 속력은 [v_{left} 'type' v_{right}]로 정의한다. type은 가속(<), 감속(>),등속(=)의 조합으로 {<, >, =, <= ,>= ,=> ,=<, <>}이다. 머신의 속력은 type이 <인 경우 속력 v_{left}에서 v_{right}로 가속하며, >인 경우 속력 v_{left}에서 v_{right}으로 감속하며, =인 경우에는 머신의 속력은 속력 v_{left}에서 v_{right}사이의 등속이고, '<>'인 경우 속력 v_{left}에서 v_{right}사이이다. 한편, 속력은 최소 0이상이며, 머신의 최대 이동 속력보다 작거나 같다. □

정의 3.5 (이동) 머신 M이 점P_i에서 점P_j로 아무런 경로를 거치지 않고 이동하는 것을 $M:\{P_i \rightarrow P_j\}$로 표기한다. 머신이 이동할 때 이동 경로가 주어진 경우 '→'대신 '⇢'로 표기한다. ({path_list}, [lower_bound, ⟨delay, period, execution⟩, upper_bound], [v_{left} 'type' v_{right}])의 형태로 머신이 이동할 때 가지는 경로, 시간, 속도에 대한 제약사항을 표현한다. □

이동조건 / 이동결과	Rule 1. (Move)	Rule 2 (No-Move)
	$\frac{M:L(M)=P_i}{M:\{P_i \rightarrow P_j\}}$	$\frac{M:L(M)=P_i}{M:\{P_i \rightarrow \neg P_j\}}$

Rule 3. (Select-Move)

$$\frac{M:\{(P_i \rightarrow P_j)\,|\,(P_i \rightarrow P_k)\},L(M)=P_i}{M:\{P_i \rightarrow P_j\}\,|\,M:\{P_i \rightarrow P_k\}}$$

Rule 4. (Via-Point)

$$\frac{M:\{P_i \rightarrow P_j, P_j \rightarrow P_k)\},L(M)=P_i}{M:\{P_i \xrightarrow{P_j} P_k\}}$$

Rule 5. (No-Via-Point)

$$\frac{M:\{P_i \rightarrow \neg P_j, P_i \rightarrow P_k)\},L(M)=P_i}{M:\{P_i \xrightarrow{\neg P_j} P_k\}}$$

Rule 6. (Cycle)

$$\frac{M:\{P_i \rightarrow P_j, P_j \rightarrow P_i)\},L(M)=P_i}{M:\{P_i \xrightarrow{P_j} P_i\}}$$

Rule 7. (In-Region)

$$\frac{M:\{P_i \rightarrow P_R, P_i \notin P_R\},L(M)=P_i}{M:\{P_i \rightarrow P_R\}}$$

Rule 8. (Not-In-Region)

$$\frac{M:\{P_i \rightarrow \neg P_R, P_i \notin P_R\},L(M)=P_i}{M:\{P_i \rightarrow \neg P_R\}}$$

Rule 9. (Out-Region)

$$\frac{M:\{P_R \rightarrow P_j, P_j \notin P_R\},L(M)=P_i, P_i \in P_R}{M:\{P_R \rightarrow P_j\}}$$

Rule 10. (Via-Region)

$$\frac{M:\{P_i \rightarrow P_R, P_R \rightarrow P_j)\},L(M)=P_i}{M:\{P_i \xrightarrow{P_R} P_j\}}$$

Rule 11. (No-Via-Region)

$$\frac{M:\{P_i \rightarrow \neg P_R, P_i \rightarrow P_j)\},L(M)=P_i}{M:\{P_i \xrightarrow{\neg P_R} P_j\}}$$

<표 1> 이동 규칙 - 기본

4. 이동 규칙

4.1 단일 머신의 이동

단일 머신이 이동할 때, 이동의 종류는 1) 아무런 제약사항 없이 이동하는 경우, 2) 시간 제약에 따라 이동하는 경우, 3)이동시 속력을 제한하는 경우, 4) 시간과 속력 모두 제한하는 경우이다. 여기에서 시간과 속력의 제약사항은 정의 3.3과 정의 3.4에 의해 표현된다.

4.1.1 이동 규칙 - 기본

머신이 이동에 관한 시간이나 속력에 대한 제약 사항 없이 이동할 때, 발생할 수 있는 이동의 규칙은 <표1>과 같다. $L(M)$은 머신 M의 위치를 나타내는 함수이다.

Rule 1은 머신 M이 현재 P_i에 위치할 때, P_i에서 P_j로 이동하는 것을 나타낸다. Rule 2는 머신M이 P_i로부터 P_j로 이동할 수 없음을 의미한다. Rule 3은 머신M이 P_i에서 여러 경로가 존재하는 경우로, 머신은 선택적인 이동을 할 수 있다. Rule 4는 머신 M은 특정 지점을 경유해야만 함을, Rule 5는 특정 지점을 경유하지 말아야 하는 제약사항을 표현하고 있다. Rule 6은 머신 M의 이동이 주기 형태를 가진 것을 나타내며, Rule 7은 머신 M이 영역R로의 진입을, Rule 8은 머신 M이 영역R에 진입 불가, Rule 9는 머신M이 영역R에 영역이 아닌 위치로의 이동을 의미한다. Rule 10은 머신 M이 영역 R을 경유해야만 하는 이동을, Rule 11은 영역 R을 경유하지 말아야 하는 이동을 나타낸다.

4.1.2 이동 규칙 - 시간

이동에 관한 시간 제약에 따라 발생하는 이동 규칙은 <표1>에 정의 3.3과 같은 시간 속성을 추가한 것이다. <표2>는 시간 제약에 따른 이동 규칙 중 일부분으로, 시간τ는 머신이 이동을 할 때 소요되는 최대 시간이다.

Rule 2.

$$\frac{M:L(M)=P_i,\{lower,\langle -,-,-\rangle,upper\}}{M:\{P_i \xrightarrow{\{lower,\langle -,-,-\rangle,upper\}} P_j\}}$$

$$\frac{M:\left\{\begin{matrix}P_i \xrightarrow{\{lower1,\langle -,-,-\rangle,upper1\}} P_j,\\ P_i \xrightarrow{\{lower2,\langle delay,-,-\rangle,upper2\}} P_j\end{matrix}\right\},\left(\begin{matrix}L(M)=P_i,\\ upper2 < lower1,\\ \tau \le (upper2-delay)\end{matrix}\right)}{M:\{P_i \xrightarrow{\{lower2,\langle delay,-,-\rangle,upper2\}} P_j\}}$$

$$\frac{M:\left\{\begin{matrix}P_i \xrightarrow{\{lower1,\langle -,-,-\rangle,upper1\}} P_j,\\ P_i \xrightarrow{\{lower2,\langle delay,-,-\rangle,upper2\}} P_j\end{matrix}\right\},\left(\begin{matrix}L(M)=P_i, delay > upper1,\\ (upper2-delay) \ge \tau\end{matrix}\right)}{M:\{P_i \xrightarrow{\{lower,\langle delay,-,-\rangle,upper2\}} P_j\}}$$

Rule 4

$$\frac{M:\left\{\begin{matrix}P_i \xrightarrow{\{lower1,\langle delay,-,-\rangle upper1\}} P_j,\\ P_j \xrightarrow{\{lower2,\langle delay,-,-\rangle upper2\}} P_k\end{matrix}\right\},L(M)=P_i}{M:\{P_i \xrightarrow{(\{P_j\},\,[low\ er1,<delay,-,->,-])} P_k\}}$$

Rule 6.

$$\frac{M:\left\{\begin{matrix}P_i \xrightarrow{\{lower1,<delay1,-,->,upper1\}} P_j,\\ P_j \xrightarrow{\{lower2,<delay2,-,->,upper2\}} P_i\end{matrix}\right\},\begin{matrix}L(M)=P_i,\\ period=p1,\\ execution=e1\end{matrix}}{M:\{P_i \xleftarrow{(\{P_j\},\,[lower1,<delay1,execution,period>,-])} P_i\}}$$

<표 2> 이동 규칙 -시간

Rule 2는 머신 M이 제한 시간에 이동을 할 수 없다면, 제한 시간 이전이나 이후에 머신이 이동을 할 수 있음을 표현한다. Rule 4는 머신 M이 특정 지점을 경유하여 갈 때의 시간은 초기 이동 시간만을 알 수 있음을 나타낸다. Rule 6은 머신의 이동이 주기 형태를 가질 때, 주기와 이동 소모 시간을 제한하여 머신이 일정 주기마다 이동 소모 시간동안 이동함을 나타낸다.

4.1.3 이동 규칙 - 속력

머신의 이동에 속력 제약을 할 경우, 머신이 이동하는 경우에만 적용되며, 이동을 하지 못한 경우에는 적용할 수 없다. 즉, <표1>에서 머신이 이동 불가능한 경우를 제외한 모든 이동에 정의 3.5에 의해 속력 제약을 한다. 만일 머신이 이동시 명세된 속력 제약을 위반하는 경우, 머신에 *velocity_error_msg*를 보내 속력 제약을 지키도록 한다. 한편, 시간 제약에 따른 이동 규칙과 마찬가지로 속력 제한에 따른 이동규칙은 <표1>과 유사하게 속력 제약을 추가한 것과 유사하나 몇몇 규칙은 <표3>와 같다.

Rule 4.

$$M:\left\{\begin{array}{l} P_i \xrightarrow{[v_{left}1'type1'v_{right}1]} P_j, \\ P_j \xrightarrow{[v_{left}2'type2'v_{right}2]} P_k \end{array}\right\} \begin{array}{l} L(M)=P_i, \\ type=type1+type2 \end{array}$$

$$\frac{}{M:\{P_i \xrightarrow{\{P_j\},\ [v_{left}1\ 'type'\ v_{right}2]} P_k\}}$$

<표 3> 이동 규칙 - 속력

4.1.4 이동 규칙 - 시간과 속력

머신의 이동에 시간과 속력 제약 모두를 적용하는 경우, 시간 제약에 의한 이동에서 이동하지 못하는 경우를 제외한 이동에 속력 제약을 추가한다.

4.2 다수 머신의 이동성

여러 머신들이 이동할 때, 각각의 머신들은 충돌을 방지하기 위해 자신만의 일정한 범위의 영역을 가지게 된다. 이런 영역은 보호 영역(*protected region*)이라 하고, 보호 영역밖에 일정 범위의 주의 영역(*alert region*)이 존재한다. 머신의 주의 영역에 다른 머신이 침범하는 경우 머신들은 충돌 가능성을 갖게 되며 보호 영역에 침범하는 경우 머신은 충돌하게 된다.

머신 간의 충돌을 방지하기 위해서 머신은 자신의 주의 영역을 다른 머신이 침범한 경우 일정 주기의 시간마다 침범한 머신과의 거리와 속력을 계산하며, 침범한 머신과의 거리가 보호 영역보다 크도록 속도, 시간을 이동의 제약사항 범위 안에서 변경한다. 또한 충돌할 만한 지점과 침범한 머신의 이동 방향을 예측하여 그 지점을 경유하지 않도록 하는 방법이 있으며, 이동 규칙을 <표4>와 같이 기술할 수 있다.

5. 예제 : 철도 건널목 시스템(Rail-road Crossing System)

지금까지 정의한 머신의 이동을 철도 건널목 시스템에 적용하면 <표 5>와 같다. <표5>는 기차가 건널목의 일정 거리(P_l) 내에 진입한 직후 차단기(*Gate*)를 5 단위 시간이내에 *up*에서 *down*으로 이동시키며(①), 기차가 완전히 지나간 후에 *down*에서 *up*으로 이동시킨다(②). 차는 5 단위 시간이내에 건널목을 지나며, 기차는 P_l에 진입한 후 최대 속력으로 5 단위 시간이상의 시간이 지난 후 건널목 영역(P_k)에 들어온다. 그리고 차는 차단기가 *down*인 경우와 이동 중일 때 건널목(P_c)을 지날 수 없으며(③), *up*인 위치에 있을 때 건널목을 건널 수 있다(④).

6. 결론 및 향후연구

본 논문에서는 이동 분산 실시간 시스템을 분석하기 위해 고안한 DATM의 이동에 대해 정의하였다. 머신의 위치, 시간, 속력 등의 속성에 제약을 둔 이동 규칙에 대해 정의하였다. 또한 다수 머신이 이동할 때 발생할 수 있는 사건을 기술하였고 충돌과 같은 문제점이 발생시 충돌을 방지하기 위한 방법들에 간략히 살펴보았다.

$$M_1:\{P_i \xrightarrow{P_k} P_j\}$$

$$M_2:\{P_m \xrightarrow{P_k} P_n\}$$

$$\frac{}{\begin{array}{l}(M_1:\{P_i \xrightarrow{\neg P_k} P_j\} \wedge M_2:\{P_m \xrightarrow{P_k} P_n\}) \\ \vee(M_1:\{P_i \xrightarrow{P_k} P_j\} \wedge M_2:\{P_m \xrightarrow{\neg P_k} P_n\})\end{array}},$$

$$M_1:\{P_i \xrightarrow{P_k} P_j\},(Time(M_1,P_k)=[a,b])$$

$$M_2:\{P_m \xrightarrow{P_k} P_n\}$$

$$\frac{}{\begin{array}{l}(M_2:\{P_m \xrightarrow{[-,<-,-,->,a-a]} P_k, P_k \xrightarrow{[a-a,<-,-,->,-]} P_n\}) \\ \vee(M_2:\{P_m \xrightarrow{[b-t,<-,-,->,-]} P_k, P_k \longrightarrow P_n\})\end{array}}$$

<표 4> 이동 규칙 - 다수 머신

① $$\frac{Train:\{P_l \rightarrow P_1\}(P_l \notin P_R, P_1 \in P_R)}{gate:\{P_{up} \xrightarrow{[0,<-,-,->,5]} P_{down}\}}$$

② $$\frac{Train:\{P_R \rightarrow P_l\}(P_j \notin P_R)}{Gate:\{P_{down} \xrightarrow{[0,<-,-,->,5]} P_{up}\}}$$

③ $$\frac{(L(Gate)=P_{down}) \vee (Gate:\{P_{up} \leftrightarrow P_{down}\})}{Car:\{P_m \rightarrow \neg P_c\}}$$

④ $$\frac{L(Gate)=P_{up}}{Car:\{P_m \xrightarrow{(\{P_c\},\ [0,<-,-,->,5])} P_n\}}$$

<표 5> 철도 건널목 시스템

향후연구는 아직 정의되지 않은 이동을 추가하여 정의하고 DATM을 이용한 검증에 대한 연구를 계속하는 것이다. 또한, DATM의 이동성을 표현하고 검증할 수 있는 모의실험을 하는 것이다.

[참고 논문]

[1] 이철, 이문근, 조기환, *실시간 시스템의 신뢰도 향상을 위한 확률명세 및 실행예측 분석 방법*, 한국 정보과학회 논문지 게재 예정

[2] 노경주, 박지연, 이문근, *추상 시간 기계를 이용한 순환 공학 정형 기법*. 한국정보과학회 소프트웨어공학회지. 제13권 제1호. pp. 32-49. 2000.

[3] C. Tomlin, G. Pappas, and S. Sastry. *Conflict resolution for air traffic management : A study in multi-agent hybrid systems*. IEEE Transactions on Ave, Cambridge, MA 02139, USA, August 1990.

[4] Cesar Munoz, Ricky W. Butler, Victor A. Carreno and Gilles Dowek, *On the Formal Verification of Conflict Detection Algorithms*, NASA/TM-2001-210864, May 2001, pp. 57

[5] Bakker, G. J. and H. A. P. Blom, *Air Traffic Collision Risk Modeling*, Proceedings of the 32nd IEEE Conference on Decision and Control. Vol. 2, San Antonio, TX, December, 1993.

[6] Kuchar James K., Yang Lee C., *Survey of Conflict Detection and Resolution Methods*, AIAA Guidance, Navigation and Control Conference, New Orleans, Aug. 97