論文97-34C-5-10

사건 기반 시간 추론 기법

(An Event-Based Temporal Reasoning Method)

李 鍾 賢 *, 李 珉 錫 *, 禹 永 運 *, 朴 忠 植 **, 金 在 熹 *

(Jonghyon Yi, Min Suk Lee, Young Woon Woo, Choong-Shik Park, and Jaihie Kim)

요 약

기존의 전문가 시스템에서는 시간 정보를 처리하기 위한 구조가 정의되어 있지 않고, 시간을 명시적으로 표현할 수 있는 규칙 표현 방법이 정의되어 있지 않기 때문에 시간에 따라 변화하는 상황에 대한 추론이 어렵다. 기존의 시간 추론 전문가 시스템들에서는 시간에 따라 변화하는 데이터의 변화 양상에 의한 시간 추론 기능들만이 구현되어, 돌발적인 사건이 발생하여 상황을 변화시키거나 사건이 발생한 시간 사이의 관계에 의해 추론해야하는 문제에는 적용할 수 없다. 이 논문에서는 시간의 흐름에 따라 연속적으로 변화하는 데이터에 의한 추론이 가능할 뿐 아니라 예상할 수 없는 시간에 발생하는 사건을 처리하고, 사건과 시간에 관련된 지식을 표현하고 표현된 지식을 이용해 추론할 수 있는 방법을 제시하였다. 제안된 사건 기반 시간 추론 기법과 이미 구현되어 있는 추론 엔진 NEO를 이용하여 시간 추론 엔진 NEO/Temporal을 구현하였으며, 제안된 사건 기반 시간 추론 기법의 장점을 보이기 위하여 공중 작전 상황 판단 및 결정 지원 시스템을 구성하고 실험하였다.

Abstract

Conventional expert systems have difficulties in the inference on time-varing situations because they don't have the structure for processing time related informations and rule representation method to describe time explicitely. Some expert systems capable of temporal reasoning are not applicable to the domain in which state changes happen by unpredictable events that cannot be represented by periodic changes of data. In this paper, an event based temporal reasoning method is proposed. It is capable of processing the unpredictable events, representing the knowledge related to event and time, and infering by that knowledge as well as infering by periodically time-varing data. The NEO/Temporal, an temporal inference engine, is implemented by applying the proposed temporal reasoning method to the NEO, an already implemented inference engine. As one of airforce applications, a situation assessment and decision supporting system is implemented to show the benefits of the proposed temporal information processing model.

* 正會員, 延世大學校 電子工學科

(Dept. of Electronic Engineering, Yonsei University)

** 正會員, 永同工科大學 電子計算學科

(Dept. of Computer Sci., Young Dong Institute of Technology)

※ 이 연구는 과학 재단의 핵심 기초 연구("전문가 시스템에서의 시간 추론에 관한 연구") 지원으로 이루어졌음.

接受日字:1996年11月25日, 수정완료일:1997年5月10日

I. 서 론

실세계에서 발생하는 사건은 시간축 상에서의 상태적, 공간적 변이로 정의할 수 있다. 그러므로 사건에 대한 지식 표현은 시간 표현을 필연적으로 포함하게 되며 변화하는 상황에서 발생하는 문제를 지능적으로 해결하기 위해서는 이와 같은 시간 표현을 포함하는 지식 표현 방법과 이 지식 표현을 사용할 수 있는 지능적 문제 해결 방법이 필요하다. 시간 추론은 시간 정보를 사용하여 시간에 대해 변화하는 상황에 대해 추론하는 것을 말한다[1] [2].

시간 추론에 대한 기존의 연구로는 논리를 확장하여 시간을 표현할 수 있도록 하는 수학적 접근 방법[3][4][5]과 시간 추론 기능을 전문가 시스템에 적용한 연구가 있다[6][7]. 시간 논리(temporal logic)를 사용하는 수학적 접근 방법은 사건이 발생한 시간을 이용하여, 여러 사건 사이에 논리적 일관성(consistency)이 성립하는가를 밝히고자 하는 것이며, 시간 추론 기능을 전문가 시스템에 적용하는 방법은 시간 추론을 이용하여 실세계의 문제를 해결하려는 것이다. 시간 추론 기능이 첨가된 전문가 시스템을 시간 추론 전문가 시스템이라고 부르며, 지금까지 발표된 대표적인 것으로서 우주 항공 분야에 적용된 WHEELS[6]와 항공기의 자동 제어에 적용된 REX[7]가 있다. WHEELS는 일정 시간 간격으로 계속 입력되는 데이터를 저장하고 데이터의 변화 양상을 관찰하여 값의 상승, 하강, 최대, 최소등을 기반으로 추론하는 것을 시간 추론으로 정의하였고, 사건-비주기적으로 발생하는 돌발 사건-의 개념을 이용한 시간 추론을 정의하지는 않았다. REX는 규칙의 구조를 (1) 시간과 사건을 다루지 않는 일반적인 규칙, (2) 사건 발생후 일정 시간이 지난 다음의 상태를 확인하는 시간 동기 규칙(Clock-synchronized rules), (3) 사건이 발생했을 때 이전 시간 구간동안의 데이터 변화 양상을 확인하는 사건 구간 규칙(Event spanning rules), (4) 주기적으로 이전 시간 구간 동안의 데이터 변화 양상을 확인하는 시간 구간 규칙(Time-spanning rules)의 4가지로 구분하여 사건과 시간을 고려하는 시간 추론을 정의하였다. REX의 경우에는 주기적으로 입력되는 데이터를 관리할 뿐 아니라 예상할 수 없는 시간에 발생할 수 있는 사건에 의해 추론이 구동되는 규칙이 있어서 WHEELS보다 확장된 시간 추론을 제시하고 있다. 그러나 REX에서의 사건은 추론의 시작을 위한 조건으로서만 사용되며 사건 사이의 선후 관계와 인과 관계를 규칙으로 표현할 수 없다.

이 논문에서는 (1) 주기적으로 입력되는 데이터를 이용하는 추론이 가능하고, (2) 사건의 발생에 의해 시작되는 추론이 가능하며, (3) 사건들이 발생한 시간의 선후 관계와 사건의 인과 관계를 표현하고 추론할 수 있으며, (4) 사건과 시간이 복합적으로 관련된 지식을 표현하고 그를 통해 추론할 수 있는 사건 기반 시간 추론 기법을 정의하고 이미 개발된 추론 엔진인 NEO에 시간 추론 기능을 첨가한 추론 엔진 NEO/Temporal을 구현하였다.

제안한 사건 기반 시간 추론 기법을 실험하기 위하여, NEO/Temporal을 공중 작전 상황 판단 및 결정 지원 시스템에 적용하여 실험하였다. 이것은 예상할 수 없는 시간에 발생하는 공중 작전 상황의 변화(사건)에 대하여 일련의 사건들과 연속적으로 입력되는 데이터에 의해 현재 상황에 대한 판단을 내리고 지휘관의 결정을 지원하기 위한 것이다.

II. 시간 추론의 필요 기능

시간 추론은 시간 속성을 가지는 상태의 변화에 대한 추론이다. 시간을 두고 발생하는 상태의 변화와 공간적 이동은 사건으로 정의되며, "언제 발생한 무엇", 즉 시간 정보와 사건의 이름으로 표현된다. 시간 정보는 정확한 순간을 나타내는 시각과 유지된 시간을 나타내는 시간 길이, 지정된 시작 시간부터 유지된 시간을 표현하는 시간 간격, 그리고 주기적인 시간을 위한 주기 표현 등으로 표현할 수 있다. 사건이 발생한 시간을 사용하여 추론하기 위해서는 사건의 이름으로 사건 발생 시간을 참조할 수 있는 시간 사건 통합 표현이 필요하다. 시간 종속 데이터는 시간축 상에서 연속적으로 입력되는 자료들로서, 각각의 값은 그 시간에서의 상태를 나타내고, 값의 변화 양상은 사건을 발생시킨다. 예를 들어, 온도 데이터는 측정 시간마다 값을 가지게 되어 그 시간에서의 온도값을 표현하며, 온도 데이터의 변화 경향인 온도의 상승이나 온도의 하락 등은 사건으로서 의미를 가진다. 시간 추론은 이러한 시간 종속 데이터에 대한 표현과 추론을 포함해야 한다. 시간 추론에서의 규칙은 사건의 표현, 시간의 표현, 시간 종속 데이터의 표현을 사용하여 시간에 따른 변화에 대한 지식을 표현할 수 있어야 한다. 그러므로 시간 추론을 위한 규칙 표현 방법이 새롭게 제시되어야하며, 여기에는 시간과 사건, 그리고 시간 종속 데이터 표현이 포함되고, 이 규칙 표현 방법을 이용한 추론 방법도 정의된다.

사건을 기반으로하는 시간 추론을 위해서는 사건 발생 시간 사이의 관계를 이용한 추론 기능과 시간 종속 데이터의 변화 경향으로부터 사건을 발생시킬 수 있는 정보 추출 기능이 필요하며, 사건의 발생으로 인해 추론이 중지되거나 시작될 수 있어야 한다. 또한 추론의 자료가 될 수 있는 사건과 시간 종속 데이터의 입력

기능이 필요하며, 이들을 효과적으로 저장, 관리하는 구조가 요구된다. 시간 추론에 필요한 표현과 기능들을 표 1에 정리하였다.

표 1. 시간 추론의 표현과 기능들.
Table 1. Representations and Functions in Temporal Reasoning.

표현측면	시간 표현	▸ 시각 (time point) 표현 ▸ 시간 길이 (time duration) 표현 ▸ 시간 간격 (time interval) 표현 ▸ 주기 (time cycle) 표현
	사건 표현	▸ 사건 이름 표현 ▸ 시간 사건 통합 표현
	시간 종속 데이터 표현	▸ 시간 종속 데이터의 종류와 주기
	규칙 표현	▸ 위의 각 표현들을 위한 규칙 구조
기능측면	추론 기능	▸ 사건 발생 시간 관계를 이용한 추론 ▸ 시간 종속 데이터로부터 정보 추출 ▸ 인터럽트에 의한 추론 시작, 중지
	입력 기능	▸ 시간 종속 데이터와 사건 입력, 저장

Ⅲ. 제안한 시간 추론 엔진 NEO/Temporal

1. NEO/Temporal의 구조와 동작

제안한 시간 추론 전문가 시스템 NEO/Temporal은 그림 1과 같은 구조를 갖는다. NEO/Temporal은 추론엔진인 NEO와 사건과 시간을 다루기 위한 모듈(사건 처리기, 시간 처리기)과 시간 참조를 위한 시스템 시계, 외부의 데이터를 입력하는 모듈(데이타 서버)을 가진다. 또한 지식 베이스로서 NEOBase는 실제 시스템의 구현에 필요한 적용 분야의 모든 규칙과 사실, 사건과 시간 데이터를 저장한다. 추론 엔진 NEO는 패턴 매칭에 의한 추론을 행한다. 시스템 시계는 시뮬레이터의 시간과 동기를 맞추기 위한 것으로 현재 시간을 표시한다. 사건 처리기는 사건의 발생을 확인하고 사건에 의한 추론을 수행하고 시간 처리기는 시스템 시계의 현재 시간과 시간 테이블의 시간을 비교하여 시간에 의한 추론을 수행한다. 데이터 서버는 시뮬레이터로부터 외부 세계의 사건과 데이터를 입력받아 NEOBase에 저장하는 부분이다. 시스템 스케줄러는 전체 흐름을 제어한다.

NEO/Temporal은 그림 2의 순서로 동작한다. 시스템 초기화 과정에서 시스템 시계를 초기화하고 NEOBase를 읽어서 시간 테이블과 사건 테이블, 시간 데이터 테이블을 작성하는 작업을 한 다음 동작 주기 동안 데이터 입력, 추론, 사건/시간 처리와 시스템 관리 과정을 반복한다. 표 2는 각 단계를 설명한다.

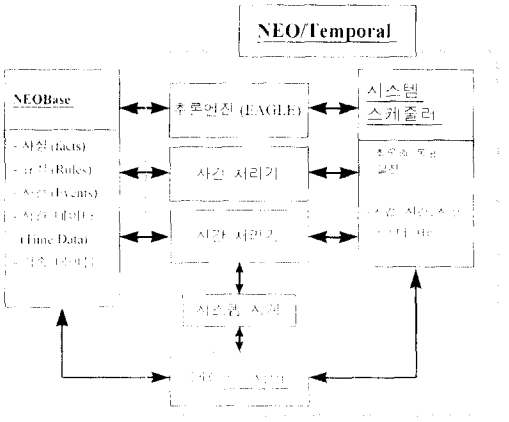


그림 1. NEO/Temporal의 구조
Fig. 1. Structure of the NEO/Temporal.

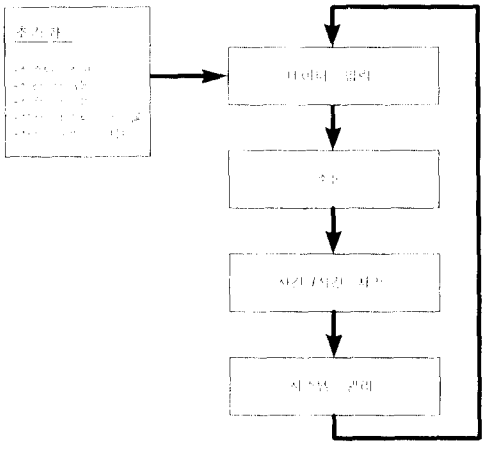


그림 2. NEO/Temporal의 동작
Fig. 2. The Flow Chart of the NEO/Temporal.

2. NEO/Temporal의 시간 관련 정보 표현

NEO/Temporal은 추론 엔진인 NEO를 사용하여 추론하기 때문에 NEO의 표현 방법을 개선하여 규칙을 표현한다. NEO의 표현은 문자열 형태로 규칙을 표현하며 그 예는 다음과 같다.

(rulename **if** <NEO 형식><NEO 형식>...)
(rulename **then** <NEO 형식><NEO 형식>...)

<NEO 형식>은 문자열로 표현되어 매칭에 사용될 수 있는 사실이나 어떤 값으로 대체되는 함수가 될 수

있다. 다음은 NEO/Temporal에서 사용되는 시간 관련 정보 표현 방법들에 대한 설명이다.

표 2. NEO/Temporal의 동작 단계 설명
Table 2. Description of Operations of the NEO/Temporal.

과정	작업
초기화	시스템 시계를 초기화하고, EAGLEBase를 참조하여 시간 테이블, 사건 테이블, 시간 데이터 테이블, 사건 발생 테이블을 작성한다.
데이터 입력	외부 세계로부터 데이터를 획득한다. 시간 데이터와 사건들을 입력받는다. 추론에 의해 결정된 명령을 외부 세계에 전해 준다.
추론	전향 추론 방법으로 추론한다.
사건/시간 처리	데이터 입력에서 입력되거나, 추론에 의해 생성된 사건에 의해 구동되는 규칙이 있는지 검사한다. 현재 시간에 관련된 규칙이 있는지 검사한다. 관련 규칙을 수행한다.
시스템 관리	사건/시간 처리에서 발생한 추론의 목표를 결정한다. 데이터 입력에서 입력된 시간 데이터를 저장한다.

1) 시간의 표현

NEO/Temporal에서는 시간 객체로서 *시각(time point)*과 *시간 길이(time duration)*를 사용한다. 시각은 숫자와 영문자 Y, M, D, h, m, s를 사용하고, 시간 길이는 시각과 같은 표현 방법에 좌우 꺾쇠 문자([,])를 덧붙여 표현한다. 시간 객체를 사용하여 시간 추론에서 사용되는 기본 단위인 *시간 간격(time interval)*과 시각을 표현한다. 시간 간격은 시작 시각과 유지 시간을 사용하여 나타내거나 시작 시각과 끝 시각을 써서 나타낸다. 주기를 가지는 시간 형태는 시작 시간과 주기를 써서 표현한다. 주기를 표현하기 위해서 "++" 기호를 사용한다. 시간 표현의 예는 표 3과 같다.

표 3. 여러 가지 시간 표현 예
Table 3. Examples of Time Representation.

	표현	의미
시각	7h30m20s	7시 30분 20초 정각
시간길이	[2h30m]	2시간 30분
시간간격	(+ 9h30m [2m])	9시 30분 부터 2분 동안
	(between 0m 3m)	0시 0분부터 3분 동안
시간주기	(++ 9h [1h])	9시부터 1시간마다
사건발생시간	(time-of <사건>)	<사건>이 발생한 시간

사건의 발생 시간을 이용한 추론을 위해서 사건이 발생한 시간을 간단한 표현으로 얻을 수 있도록 하는 time-of라는 시제 연산자를 정의한다. "경고 램프가 켜짐"이라는 사건의 이름을 (Warning Light On)이라고 약속할 때, 경고 램프가 켜진 시간이 9시부터 10시까지라면 (time-of (Warning Light On)) 은 (+ 9h [1h]) 과 같은 시간을 나타낸다.

시간 종속 데이터가 규칙에서 사용될 때에는 시간 종속 데이터의 이름과 데이터의 근원, 그리고 참조할 시간 지정을 사용하여 표현한다. 표 4는 사건과 시간 종속 데이터의 표현의 예이다. 시간 종속 데이터를 참조하는 것은 항상 과거의 값이므로 참조 시간 지정은 "현재로부터 지정 시간 전"의 의미를 갖는다.

표 4. 사건과 시간 종속 데이터의 표현 예
Table 4. Examples of Event and Time Dependent Data Representation.

	표현	의미
사건	(flight H1 found)	H1라는 이름을 가진 비행체 출현
시간 종속 데이터	(speed H1 3m20s)	H1의 3분 20초 전의 속도 값

2) 시간 관련 규칙의 표현과 의미

시간 관련 규칙은 지정된 시간에 추론할 수 있도록 NEO 규칙의 표현에 시간 관련 표현을 첨가해야 한다. NEO/Temporal에서는 그림 3과 같이 규칙의 if-then 구조 안에 시간 지식을 표현할 수 있는 시간 지정 기호를 사용하여 시간을 표현한다. 이때, at은 규칙의 참과 거짓을 판정하는 시간을 지정해주는 표현을 규칙에 명시적으로 나타내는 역할을 하며, 이 at을 포함하는 표현을 at 표현이라고 한다.

```
표현:
(rule1 if (at (<NEO 형식 A><NEO 형식 B>)
                              <시간 지정 형식>))
(rule1 then <NEO 형식 C>)
의미: <시간 지정 형식>의 시간에,
   <NEO 형식 A>와 <NEO 형식 B>가 참이면
   <NEO 형식 C>는 참이다.
```

그림 3. "**at**"을 사용한 시간관련규칙 표현 예.
Fig. 3. Example of a Temporal Rule Using "**at**".

at 표현의 첫 번째 인수는 NEO/Temporal의 여러 형식이 될 수 있다. 여기에는 사실(fact)을 나타내는

보통의 문자열 표현((?x is-a friendbase)...), 함수 표현((+ 30 40)...), 시제 연산자 표현(BEFORE, MEET,..), at 표현이 포함되며 <NEO 형식>이라고 부른다. 두 번째 인수는 규칙이 참조되는 시간을 가리키기 위한 것으로 사건 형식((hostile-flight ?x found)...)과 시간 형식((9h00m00s)...)이 될 수 있다. at 포함하는 규칙은 at 표현의 둘째 인수에 따라서 의미가 달라지며 둘째 인수가 만족되는 시간이 되면 첫 번째 인수에 대한 동작을 시작한다.

두 번째 인수가 사건 형식인 경우에 그 규칙을 사건 구동 규칙이라고 부르며 그 의미는 "사건 A가 발생하면 첫 번째 인수인 <NEO 형식>을 수행하라"는 것이다. 첫 번째 인수 <NEO 형식>이 보통의 문자열이라면 그 문자열을 사실 리스트(fact list)에서 찾아보고, 만약 함수라면 함수를 수행하게 되며, 시제 연산자인 경우는 사건 사이의 시간을 확인하며, at 표현이라면 그것의 두 번째 인수에 따른 동작을 하게 된다.

두 번째 인수가 시간 형식인 경우에는 그 규칙을 시간 구동 규칙이라고 부르고 그 의미는 "기록된 시간에 첫 번째 인수인 <NEO 형식>을 수행하라"는 것이다. 사건 구동 규칙과 마찬가지로 시간 구동 규칙에서도 <NEO 형식>에 따라서 매칭을 하거나 함수 값을 연산하고 또는 at 표현의 두 번째 인수에 따른 동작을 하게 된다.

이러한 방법으로 시간 규칙을 표현하게 되면 복잡한 지식까지도 자유롭게 표현할 수 있게 된다. 즉 (rulename if (at (at (<NEO 형식>) <시간 지정 형식>) <시간 지정 형식>))과 같이 at 표현의 첫 번째 인수로 다시 at 표현이 나오는 경우는 시간과 사건에 대한 여러 가지 관계에 대한 복합적인 지식을 표현할 수 있다. 외부 at 표현의 두 번째 인수와 내부 at 표현의 두 번째 인수 내용에 따라서 여러 가지 의미를 가지는 규칙이 되는 것이다. 가령 외부 at 표현의 시간 지정 형식이 사건 형식 A이고 내부 at 표현의 시간 지정 형식이 시간 형식 T라면 "사건 A가 발생한 때부터 시간 T가 흐르면 <NEO 형식>에 따른 동작을 하라"는 의미가 된다.

3) 시간 관련 규칙의 동작

NEO/Temporal에서는 시간 추론을 위해서 시간 테이블과 사건 테이블에 규칙과 연관된 시간과 사건들을 부가적인 정보와 함께 기록하여 시간 관련 규칙의 구동을 가능하게 한다.

① 시간 테이블

시간에 의해 구동되는 규칙들은 규칙이 구동되는 시간과 현재 시간을 비교하여 현재 시간이 규칙이 구동되는 시간과 일치할 경우 추론을 시작한다. 시간에 의해 구동되는 규칙들의 "구동 시간", 즉 at 표현의 인수인 <시간 지정 형식>을 규칙의 이름과 함께 시간 테이블에 기록함으로써 현재 시간과 비교를 간단히 할 수 있다. 지식 베이스에 rule102라는 시간 구동 규칙이 있을 때 시간 테이블에 기록되는 내용은 표 5와 같다.

표 5. 시간 테이블의 예
Table 5. An Example of a Time Table.

규칙	(rule102 if (at (it is peacetime) (++ 9h0m0s [1h]))) (rule102 then (B1 (:send 2 128.0 37.0)))					
시간 테이블	시간내역	규칙이름	참조수준	사용여부	우선순위	주기
	9h0m0s	rule102	1	-	100	[1h]
	...	...	...	...	...	...

NEO/Temporal은 시스템 동작 준비 단계에서 지식 베이스를 탐색하여 시간에 의해 구동되는 여러 규칙인 at 표현의 두 번째 인수가 시간 지정 형식인 규칙을 찾아서 이 테이블에 기록한다. 시간 테이블에 기록되는 내용은 다음과 같다.

● 시간 내역 : 시간 구동 규칙의 at 표현의 두 번째 인수의 내용이다. 시각일 수도 있고 시간 간격일 수도 있다. 주기 형식은 시작 시간과 주기로 표현하는데 시간 테이블에 주기를 기록할 때에는 주기의 시작 시간을 *"시간 내역"*에 기록하고 주기를 *"주기"*에 기록한다.

● 참조 수준 : 규칙에 사용된 at 표현의 개수를 나타낸다. at 표현의 첫 번째 인수가 다시 at 표현을 포함하는 경우와 같이 at 표현이 여러번 사용되는 경우를 처리하기 위한 것이다.

● 사용 여부 : 시간 규칙의 사용 여부를 기록한다.

● 우선 순위 : 시스템의 한 동작 주기 안에 두 개 이상의 시간 규칙이 동작하게 될 때 우선 순위를 결정하기 위해 사용된다.

NEO/Temporal의 동작 중 시간 관련 동작 과정으로 들어가게 되면 시스템 내부 시계와 시간 테이블을 비교하여 동작해야 할 규칙을 선택하고 실행한다. 시간 구동 규칙은 NEOBase에서 at 표현의 첫 인수를 찾는다. 한 규칙에 at 표현이 여러번 사용될 경우, 첫 번째 사용된 at 표현의 시간 지정 형식이 만족되는지 확인한

다. 만족되는 경우에는 참조 수준을 "1" 만큼 감소시키고, 두 번째 at 표현의 시간 지정 형식을 확인하여 만약 시간 형식이라면 시간 지연되는 시간만큼을 현재 시간이나 "시간 내역"에 더하여 시간 테이블에 기록하고 사건 형식이라면 사건 내용을 사건 테이블에 기록한다. 이 때 규칙 이름은 동일하지만 참조 수준이 감소하게 된다. 이러한 규칙이 수행되는 시점은 참조 수준이 "1"이 되었을 때이다. 따라서, 참조 수준이 "2"이상이라는 것은 이 규칙의 시간이 만족되어도 규칙의 동작이 바로 실행되지 않는다는 것을 의미한다.

2) 사건 테이블

사건에 의해 구동되는 규칙은 외부 사건 입력이나 추론 중에 발생한 내부 발생 사건에 의해 구동되는 규칙이므로, 현재 시간에 발생한 모든 사건들을 규칙의 구동 조건이 되는 사건 이름과 비교하여 그 규칙을 구동할 것인지를 결정해야 한다. 사건에 의해 구동되는 규칙들의 구동 조건이 되는 at 표현의 두 번째 인수인 사건 이름을 규칙의 이름과 함께 사건 테이블에 기록하여 사건에 의해 구동되는 규칙을 찾는다. 지식 베이스에 사건 구동 규칙 rule103이 있는 경우 사건 테이블의 내용은 표 6과 같이 기록된다.

표 6. 사건 테이블의 예
Table 6. An Example of a Event Table.

규칙	(rule103 if (at ((?y is-target-of ?x)(?x attack-time ?time)) (hostile-flight ?x Is-found))) (rule103 then (target ?y isdetermined))			
사건 테이블	사건 이름	규칙이름	참조수준	우선순위
	(hostile-flight ?x is-found)	rule103	1	100

NEO/Temporal의 동작 준비 과정에서 NEOBase를 탐색하여 at 표현의 두 번째 인수가 "사건 이름"인 경우를 찾아 사건 테이블에 기록한다. 사건 테이블에 기록되는 내용은 다음과 같다.

● 사건 이름 : at 표현의 두 번째 인수인 "사건 이름"을 기록한다.

● 참조 수준 및 우선 순위: 시간 테이블의 경우와 같다.

NEO/Temporal의 동작 중 사건 관련 동작 과정이 되면 지난 동작 주기 동안 발생한 사건들과 사건 테이블을 비교하여 동작시킬 규칙을 선택하고 실행한다. 사건 구동 규칙은 NEOBase에서 at 표현의 첫 인수를 찾는다. 내부 at 표현이 있는 경우에는 지연되는 시간만큼을 현재 시간이나 사건 발생 시간에 더하여 시간 테이블에 기록하거나 새로운 사건 내용을 사건 테이블에 기록한다. 이 경우 앞에서 설명한 바와 같이 규칙 이름은 동일하지만 참조 수준이 감소하게 된다.

3. 시간 추론 기능

1) 사건 사이의 시간 관계에 의한 추론

기존의 시간 추론 전문가 시스템에서는 규칙을 동작시키기 위한 조건으로서 사건을 사용하였다[7]. 사건이 규칙의 동작 조건으로서만 사용될 경우에는 사건 사이의 인과 관계와 선후 관계를 규칙으로 표현하는 방법이 정의되지 않는다. 제안한 사건 기반 시간 추론 기법에서는 사건 사이의 인과 관계와 사건의 시간 간격 사이의 선후 관계를 규칙으로 표현하고 추론하기 위하여 논리적 시간 추론 방법인 Allen의 방법을 적용하였다.

Allen은 시간 간격 사이의 13가지 관계를 정의하였고 그에 의한 추론을 제안하였다[13]. 이 시간 관계 중 6 가지는 다른 6 가지 시간 관계의 역관계를 표현하므로 전문가 시스템에 지식을 구축하고 이를 통해 추론할 때에는 중복을 피해 7 가지의 관계만으로 충분하다.

사건 사이의 시간 관계를 표현하기 위해서는 지식 공학자가 사건의 발생 시각과 유지 시간에 대해 주의를 기울이지 않아도 사건의 이름만으로 발생 시각과 유지 시간을 알아내어 비교할 수 있는 7 가지 시제 연산자(modal operator)를 사용한다. 이들 가운데 BEFORE의 경우에는 첫 번째 사건의 끝 시각(xt)과 두 번째 사건의 시작 시각(yi)을 알아내어 전자가 후자보다 앞설 때(xt<yi) 참이 된다. 이와 같이 지식 공학자는 사건의 이름만을 가지고 시간 관계를 표현하고, 전문가 시스템 도구에서는 각 시간 사이의 관계에 대한 정의와 동작을 포함하고 있어야 한다. 표 7은 이들 7가지의 시제 연산자에 대한 의미를 설명하고 있다. 그림 4는 사건의 시간 관계 중 MEET 시제 연산자를 설명하는 것으로서, "미사일이 발사되었다" 라는 사건과 "적기가 사라졌다" 라는 사건의 발생 시간이 MEET 관계라면 "적기가 격추되었다" 라는 결론을 얻는다.

4. 사건과 시간 데이터의 입력과 저장

시간 추론 전문가 시스템은 끊임없이 외부 세계의 변화들을 관측하고 입력받아서 추론하고 상황에 적절한 결과를 제공해야 한다. 외부 세계의 변화들은 예측할 수 없는 시간에 발생하는 사건들과 연속적으로 변

하는 데이터로 구분할 수 있다. NEO/Temporal에서는 외부 세계의 변화 중 예측할 수 없는 시간에 발생하는 사건을 비동기적인 사건으로, 연속적으로 변하는 데이터는 시스템에서 지정한 간격으로 갱신하게 되므로 동기적인 시간 종속 데이터로 구별하여 추론의 자료로 삼는다.

표 7. 사건 사이의 시간 관계와 시제 연산자
Table 7. Temporal Relations between Events & Modal Operators.

Allen의 시간관계	시제 연산자	(time-of X)=(between xi xt) (time-of Y)=(between yi yt)
X before Y (Y after X)	BEFORE	xt < yi
X equal Y (Y equal X)	EQUAL	xi = yi xt = yt
X meets Y (Y met-by X)	MEET	xt = yi
X overlaps Y (Y overlapped X)	OVERLAP	xi < yi < xt < yt
X during Y (Y includes X)	DURING	yi < xi xt < yt
X starts Y (Y started-by X)	START	xi = yi xt < yt
X finishes Y (Y finished-by X)	FINISH	xi > yi xt = yt

표현 : (rule105 if ((MEET (missile ?x is-shoot-to ?y)
(hostile-flight ?y disappear)))
(rule105 then (hostile-flight ?y destroyed))

의미 : 적기 ?y를 향해 미사일 ?x가 발사되고 미사일이 소멸되는 시간과 적기 ?y가 소멸하는 시간이 MEET 관계이면 ?y는 ?x에 의해 파괴된다.

그림 4. 사건의 시간 관계인 MEET의 예
Fig. 4. An Example of the MEET Relation Between Events.

1) 사건의 입력과 저장

NEO/Temporal에서는 적용 분야에서 발생할 것으로 예상되는 사건들의 이름을 미리 정의한다. 정의된 사건들이란 NEOBase에 구축된 규칙들에서 사용되는 사건이다. 사건들의 이름은 표 6과 같이 사건 테이블에 기록되어 사건 구동 규칙을 구동시킨다.

실제 시스템 동작 중에는 발생하는 모든 사건을 데이터 입력 과정에서 입력받아 사건 발생 테이블에 기록하고, 사건 처리 과정에서 발생 여부를 확인하여 규칙을 구동한다.

2) 시간 종속 데이터의 입력, 저장

어떤 시간 종속 데이터들이 입력되는지는 적용 분야마다 정해지므로 NEO/Temporal에서는 시간 종속 데이터를 지식 구축 단계에서 미리 정의하여 사실이나 사건과는 구별하여 사용한다. 예를 들어 공군 작전 상황 판단 및 결정 지원 시스템에서는 레이더에 관측된 항공기의 위치와 속도들이 주기적으로 입력된다. 시간 종속 데이터마다 입력되는 주기가 다르고 데이터의 형태가 다르므로 NEO/Temporal에서는 모든 시간 종속 데이터의 입력 주기와 데이터의 형태를 미리 정한다. 시스템의 동작 중에는 데이터 입력 과정에서 그 시간에 입력되어야 하는 데이터들의 값을 데이터의 근원과 데이터 값, 입력 시간과 함께 시간 종속 데이터 저장 테이블에 저장한다.

Ⅳ. 실험 및 결과

1. 실험 내용

NEO/Temporal의 시간 추론 기능을 실험하기 위해서 공중 작전 상황 판단 및 결정 지원 시스템에 적용하였다. 실험을 위한 공중 상황 시뮬레이터에서는 적기의 이동, 아군 비행기들의 위치, 아군의 현재 상태 등 현재의 모든 상황을 발생시킨다. 발생된 상황에 관련된 정보들은 NEO/Temporal의 입력 정보로 사용되어, 상황 평가와 대응 방안을 수립하게 된다. 대응 방안 결과는 다시 시뮬레이터로 보내져 수행된다. 시뮬레이터에서 입력되는 정보는 현재 시간의 비행기 위치 정보와 속도 정보, 그리고 사건의 이름이다.

예측할 수 없는 시간에 발생하는 사건에 의한 추론과 예측된 시간 간격 사이의 관계에 의한 추론을 실험하기 위해 간단한 실험 시나리오를 사용하였다. 시나리오는 실제 공군 작전 상황에서 적군기가 자신의 의도를 감추기 위해 비행 경로를 수차례 수정하는 경우 발생하는 여러 사건과 데이터의 변화를 고려한 것이다.

2. 실험 결과

NEO/Temporal은 NEOBase를 탐색하여 사건 구동 규칙과 시간 구동 규칙을 찾아 사용되는 사건들을 표 8과 같이 사건 테이블에 기록한다. 사건 테이블은 적기의 예상 공격 목표 결정과 예상 공격 시간 결정을 위한 규칙, 적기의 방향 전환 감시를 위한 규칙, 대응 방안 결정을 위한 규칙에서 사건에 의해 구동되는 규

칙을 찾아내어, 사건의 이름과 관련된 규칙과 부가 사항을 함께 기록한다.

시뮬레이션을 시작하면 임의의 시간에 침투하는 적기가 아군 레이더에 의해 발견된 시각부터 상황이 시작된다. 그림 8의 (a)는 적기가 아군 레이더 기지에 의해 포착되는 시점의 상황을 나타낸다. 이 때 입력되는 정보는 적기가 발견되었다는 사건 (hostile-flight H1 is-found)와 주기적으로 입력되는 시간 데이터인 적기 H1의 위치, 속도 정보이다. 적기 발견 사건에 의해 NEO/Temporal에서는 rule01 규칙을 구동하여 적기의 공격 목표((Radar2 is-target-of H1))를 찾는 추론과 예상 공격 시간((H1 attack-time 09h31m40s))을 예측하는 추론, 그리고 rule21을 구동하여 공격 목표와 예상 공격 시간에 따른 방어 방법((B2 can-assist Radar2))을 추론한다(그림 8의 (b)). 또한 NEO/Temporal은 입력되지 않는 정보인 방향 전환 사건을 감지하기 위해 적기의 방향을 계속 추적하게 된다. 적기의 방향을 추적하는 것은 "적기 발견"이라는 사건에 의해 구동되어 주기적으로 적기의 방향을 확인하는 규칙 rule10을 사용하여 적기의 방향 전환 여부를 확인한다.

표 8. 사건 테이블 작성
Table 8. Making Up the Event Table.

사건 이름	규칙이름	참조수준	우선순위
(hostile-flight ?x is-found)	rule01	1	100
(hostile-flight ?x is-found)	rule10	2	90
(hostile-flight ?x change-direction)	rule02	1	100
(target ?y is-determined)	rule21	1	90

시뮬레이터에 의해 적기는 진행되다가 예측할 수 없는 시간에 방향을 바꾸어 최종적인 공격 목표를 향하게 된다(그림 9의 (a)). 이때 NEO/Temporal은 적기의 방향이 바뀐 것을 규칙 rule10에 의해 확인하여 내부 발생 사건인 (hostile-flight H1 change-direction)이라는 방향 전환 사건을 발생시키고, 이 사건과 (hostile flight H1 is-found) 사건 사이의 BEFORE 관계에 의해 새로운 공격 목표를 결정하기 위한 내부 발생 사건 (determine-new-target-of hostile-flgiht H1)을 발생시킨다. 그리고 이 사건에 의해 사건 구동 규칙 rule02를 수행한다. 추론의 결과로 새로운 적기의 공격 목표가 Radar1으로 예측되고, 적기가 Radar1까지 도달하는 시간((H1 attack-time 09h53m15s))을 예측하며, 이러한 정보에 의한 새로운 방어 방법을 rule21을 사용하여 이미 파견된 아군기 B2-0와 아군기지 B1에서 방어하도록 하는 결정((B2-0 can-assist Radar1), (B1 can-assist Radar1))을 내렸다(그림 9의 (b)). 이러한 NEO/Temporal의 결정에 의해 시뮬레이터는 B1 비행장으로부터 B1-0 아군기를 새로 파견하고, 이미 파견되었던 B2-0 아군기의 방향을 Radar1으로 전환시킨다.

```
(rule01 if (at ((?y is-target-of ?x)(?x attack-time ?time))
           (hostile-flight ?x is-found)))
(rule01 then (target ?y is-determined))
(rule02 if (at ((?y is-target-of ?x)(?x attack-time ?time))
           (determine-new-target-of hostile-flight ?x)))
(rule02 then (target ?y is-determined))
(rule03 if (findtarget ?x ?y))
(rule03 then (?y is-target-of ?x))
(rule04 if ((?y is-target-of ?x)(timeto ?x ?y ?time)))
(rule04 then (?x attack-time ?time))
```

그림 5. 적기의 예상공격목표와 예상공격시간을 결정하는 규칙
Fig. 5. Rules for Deciding an Expected Target of a Hostile Flight and Expected Assaulting Time.

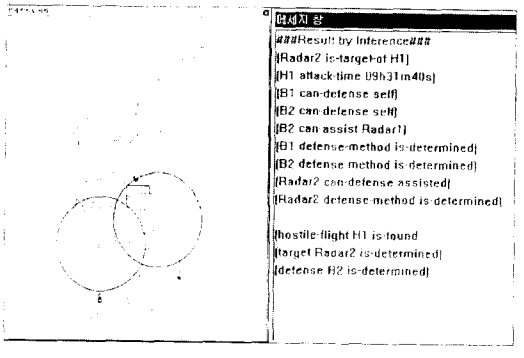
```
(rule10 if (at (at (?x direction changed) (-- (now) [4s]) )
               (hostile-flight ?x is-found)))
(rule10 then (hostile-flight ?x change-direction))
(rule11 if ((directionof ?x 0 ?dir1) (directionof ?x -1 ?dir2)
               (!= ?dir1 ?dir2)))
(rule11 then (?x direction changed))
(rule12 if ((BEFORE (hostile-flight ?x is-found)
               (hostile-flight ?x change-direction)))
(rule12 then (determine-new-target-of hostile-flight ?x))
```

그림 6. 적기의 방향 전환을 감시하는 규칙
Fig. 6. Rules for Detecting Direction Change of a Hostile Flight.

이러한 실험을 통해, 사건 구동 규칙인 rule01, rule02이 사건 발생에 의해 동작을 시작하는 것을 알 수 있으며, 사건에 의해 구동되어 4초마다 적기의 방향 변화를 감시하기 위해 at 표현이 두 번 사용된 규칙 rule10이 제대로 동작하는 것을 확인할 수 있었다. 또한 rule10에서 연속적으로 입력된 시간 데이터인 방향을 사용하여 추론할 수 있는 기능을 확인할 수 있었으며, rule12에서와 같이 사건 사이의 관계를 정의하는 시제 연산자의 동작을 확인할 수 있었다.

```
(rule21 if (at (?y defense-method is-determined)
               (target ?y is-determined)))
(rule21 then (defense ?y is-determined))
(rule22 if (?y can-defense self))
(rule22 then (?y defense-method is-determined))
(rule23 if (?y is-a friend-base))
(rule23 then (?y can-defense self))
(rule24 if (?y can-defense assisted))
(rule24 then (?y defense-method is-determined))
(rule25 if ((?y is-a friend-radar) (?z is-a friend-base)
                                  (?z can-assist ?y))
(rule25 then (?y can-defense assisted))
(rule25-1 if ((?y is-a friend-radar) (?z is-a friend-flight)
                                  (?z can-assist ?y))
(rule25-1 then (?y can-defense assisted))
(rule26 if ((?y is-target-of ?x)(?x attack-time ?time)
          (timeto ?z ?y ?btime)(<= ?btime ?time)))
(rule26 then (?z can-assist ?y))
```

그림 7. 대응 방안 수립을 위한 규칙
Fig. 7. Rules for Deciding a Defense Method.

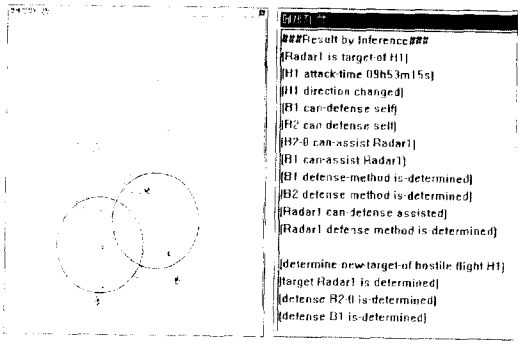


(a) (b)

그림 8. (a) 적기의 발견. (b) NEO/Temporal의 추론 결과
Fig. 8. (a) Detection of a Hostile Flight. (b) Inference Results of the NEO/Temporal.

V. 결 론

기존의 시간 추론 기법에서는 시간 추론의 범위를 연속적으로 입력되는 데이터를 이용한 추론으로 정의하거나, 사건의 입력에 의해서 시작되는 추론으로 정의하여 시간과 사건에 관한 지식을 광범위하게 표현하기 어려웠다. 그러나 NEO/Temporal은 연속적으로 입력되는 데이터와 예상할 수 없는 시간에 발생하는 사건을 이용한 시간 추론 외에도 시간 관계를 표현하기 위한 7 가지의 시제 연산자와 사건의 발생 시간과 특정 시간을 표현할 수 있는 시간 지정 기호 at을 정의하여 사건 사이의 시간 관계를 이용한 추론과 사건과 시간이 복합적으로 관련된 지식을 이용한 추론을 수행할 수 있었다.



(a) (b)

그림 9. (a) 적기의 방향 전환. (b) NEO/Temporal의 추론 결과
Fig. 9. (a) Changing Direction of the Hostile Flight. (b) Inference Results.

이 논문에서는 NEO/Temporal을 실험하기 위해 공중 작전 상황 판단 및 결정 지원 시스템을 구현하였다. 이 실험을 위해서 공중 작전 상황을 표현하고 데이터를 발생시키고 NEO/Temporal로부터의 명령을 실행할 수 있는 시뮬레이터를 사용하였으며, 미리 입력된 경로로 적기가 침투, 이동하는 상황을 가정하여 아군의 상황 판단과 대응 방안 수립 임무를 NEO/Temporal에서 수행하도록 하였다. 실험 결과 연속적으로 입력되는 데이터와 사건을 이용한 추론과 지정된 시간에 이루어지는 추론을 통해 상황을 판단하고 대응 방안을 수립할 수 있었다.

이 논문에서 제안한 시간 추론 기법이 실세계의 복잡한 문제에 적용되기 위해 연구되어야 할 과제는 다음과 같다. 첫째, 상황 평가와 결정 지원, 공장 제어, 로봇 제어 등 실시간 처리를 필요로 하는 응용 분야에서의 시간 추론은 실시간에 입력되는 정보를 바탕으로 실시간에 결론을 보여야 하므로 실시간 처리가 가능한 시간 추론 기법이 요구된다. 둘째, 시간 추론 전문가 시스템에 적합한 불확실성의 표현이다. 기존의 전문가 시스템에서는 지식의 불확실성을 표현하기 위해 확률이나 확신률 등의 방법을 사용하는데, 시간 추론 전문가 시스템에서의 사건과 시간에 관련된 지식을 규칙으로 표현할 때에는 시간이라는 새로운 개념적 범주에

알맞은 불확실성 표현 방법이 연구되어야 할 것이다.

참 고 문 헌

[1] J. M. McCarthy and P. J. Hayes, "Some Philosophical Problems from the Standpoint of Artificial Intelligence," in *Readings in Artificial Intelligence*, Tioga, Palo Alto, Calif., pp.431-450, 1981.

[2] P. J. Hayes, "The Naive Physics Manifesto," in J. R. Hobbs and R. C. Moore, eds., *Formal Theories of the Commonsense World*, Ablex, Norwood, N. J., 1984.

[3] J. F. Allen, "Towards a general theory of action and time," *Artificial Intelligence* vol. 23, pp. 123-154, 1984.

[4] D. V. McDermott, "A temporal logic for reasoning about processes and plans," *Cognitive Sci.* 6, pp. 101-155, 1982.

[5] Yoav Shoham, "Temporal Logics in AI: Semantical and Ontological Considerations," *Artificial Intelligence* 33, pp. 89-104, 1987.

[6] W. A. Perkins and A. Austin, "Adding Temporal Reasoning to Expert System Building Environments," *IEEE Expert*, pp 23-30, February 1990.

[7] Bandreddi E. Prasad, Tolety Siva Perraju, Garimella Uma and Pasuparthy Umarani, "An Expert System Shell for Aerospace Application," *IEEE Expert*, pp. 56-64, August 1994.

저 자 소 개

李 鍾 賢(正會員)

1994년 2월 연세대학교 전자공학과 공학사. 1996년 2월 연세대학교 전자공학과 공학석사. 1996년 3월 ~ 현재 연세대학교 전자공학과 박사 과정. 주관심분야는 인공지능, 문서인식, 얼굴인식 등임

李 珉 錫(正會員)

1993년 2월 연세대학교 전자공학과 공학사. 1995년 2월 연세대학교 전자공학과 공학석사. 1995년 3월 ~ 현재 연세대학교 전자공학과 박사 과정. 주관심분야는 인공지능, 전문가시스템, 시간추론, 정보융합 등임.

禹 永 運(正會員)

1989년 2월 연세대학교 전자공학과 공학사. 1991년 8월 연세대학교 전자공학과 공학석사. 1992년 3월 ~ 현재 연세대학교 전자공학과 박사 과정. 주관심분야는 인공지능, 전문가시스템, 정보융합, 패턴인식 등임.

朴 忠 植(正會員)

1985년 2월 한양대학교 전자공학과 공학사. 1987년 8월 연세대학교 전자공학과 공학석사. 1991년 8월 연세대학교 전자공학과 공학박사. 1993년 ~ 현재 영동공과대학 전자계산학과 교수. 주관심분야는 인공지능, 전문가시스템, 정보융합, 객체지향시스템 등임.

金 在 熹(正會員)

1979년 연세대학교 전자공학과 공학사. 1982, 1984년 Case Western Reserve University 전기공학과 공학석사, 공학박사. 1984년 ~ 현재 연세대학교 기계전자공학부 전자공학전공 교수. 주관심분야는 전문가시스템, 정보융합 등의 인공지능과 문자인식, 서명검증, 지도인식, 얼굴인식 등의 패턴인식 분야임.