상호주도형 대화와 상황인식을 통한 지능형 일정관리 에이전트

임성수° 홍진혁 송인지 조성배
연세대학교 컴퓨터과학과
{lss°, hjinh, schunya}@sclab.yonsei.ac.kr, sbcho@cs.yonsei.ac.kr

Intelligent Schedule Management Agent with Mixed Initiative Conversation and Context-awareness

Sungsoo Lim° Jin-Hyuk Hong Injee Song Sung-Bae Cho
Dept. of Computer Science, Yonsei University

요 약

업무 능률을 높이기 위해 일정의 입력이나 변경 및 삭제를 체계적인 관리하는 일정관리 에이전트에 대한 연구가 수행되고 있다. 사용자와의 지속적인 상호작용을 통해 사용자의 의도를 파악하며, 사용자에게 적절한 일정을 추천하거나 관련된 정보를 제공하는 서비스가 함께 요구된다. 본 논문에서는 상호주도형 대화를 통해 사용자의 의도를 능동적으로 추론하여 보다 자연스러운 대화를 제공하며, 상황인식을 통해 사용자에게 적절한 일정관리 서비스를 제공하는 지능형 일정관리 에이전트를 제안한다. 대화의 문맥 유지와 상호주도적 문제해결을 위한 반응형 스크립트를 설계하고, 동적 베이지안 네트워크로 일정과 관련된 사용자의 상태를 추론한다. 사용자에게 적절한 일정시간이나 장소를 추천하기 위해 최근 활발히 연구되고 있는 상식 DB인 ConceptNet을 도입하여 지능적인 일정관리 서비스를 제공한다. 각종 시나리오를 통해 제안하는 에이전트의 유용성을 검증하였다.

1. 서론

일정관리는 업무를 효율적으로 수행하도록 회의, 약속 등에 대한 일정을 체계적으로 관리하는 서비스이며, 이를 위해서는 많은 시간과 노력이 필요하다. 최근 인간이 아닌 지능형 시스템을 도입하여 사용자의 일정을 효과적으로 관리하려는 시도가 이루어지고 있다. 실제로 Calendar Manager, Office Tracker, MS Outlook 등과 같은 각종 일정관리 응용 어플리케이션이 개발되어 있으며, 사용자로 하여금 자신의 일정을 손쉽게 추가, 수정, 삭제하도록 지원하며, 기록된 일정정보를 시각화하여 보여준다. 그러나 이러한 응용 어플리케이션은 시스템 중심의 인터페이스를 제공하여 사용자가 시스템을 충분히 이해해야 하며, 사용자들의 일정에 대한 우선순위와 선호도를 고려하지 않아 일정을 설계하는데 부족하다[1].

일정관리 에이전트가 사람과 같은 역할을 수행하기 하기 위해서 사용자와 에이전트 간의 원활한 의사소통과 사용자 성향 학습이 필요하다[2]. 또한 적절한 서비스를 제공하기 위해 각종 정보의 추천 기능이 함께 요구된다. 본 논문에서는 상호주도형 대화관리를 통해 사용자와 자연스럽게 대화하고 상황인식을 통해 사용자에게 적절한 서비스를 제공해주는 지능형 일정관리 에이전트를 제안한다. 상호주도형 대화를 위해서 이를 위한 스크립트 언어를 제안하고, 동적 베이지안 네트워크를 이용한 사용자의 피로도를 추론과 ConceptNet을 활용한 장소 추천을 통해 지능적인 일정관리 서비스를 제공한다.

2. 관련연구

2.1 상호주도형 대화

문제를 주도적으로 해결하는 주체에 따라 사용자기반, 시스템기반, 상호주도로 지능형 시스템을 구분할 수 있다. 사용자기반은 사용자가 직접 문제를 이해하고 해결하는 것으로 메뉴, 옵션, 명령어 등이 있으며, 시스템기반은 문제와 해결과정을 시스템에 미리 정의해 놓고 이에 대한 사용자 답변에 따라 동작하는 것으로 열차 예약을 위한 프레임 기술 등이 있다. 이와는 달리 사용자와 시스템 사이의 반복적인 의사소통을 통해서 함께 문제를 해결하는 상호주도형 대화기술이 활발히 연구되고 있다. 표 1은 문제해결의 주체에 따른 지능형 시스템의 특성을 보여준다.

표 1. 문제해결주체에 따른 시스템 특성

사용자기반	시스템기반	상호주도형
장점: •다양한 문제해결가능 •계산상 장점(시스템) •다양한 메뉴와 옵션 제공 단점: •효과적인 메뉴, 옵션 설계의 어려움 •사용자훈련 필요 •문제에 대한 사용자의 정확한 이해필요	장점: •사용자 의도 파악용이 단점: •미리 정의된 동작만 수행 •제한된 사용자의도 •불필요한 정보교환 발생	장점: •실세계 문제해결방법 •사용자의도파악 •점진적인 문제해결 •최소한의 정보교환 •복합적 의도표현가능 단점: •구현이 어려움

ELIZA, ALICE 등과 같은 초기의 대화 에이전트는 사용자의 단일질의에 대한 답변을 제공하는 질의/응답에 초점이 맞추어 있었다. 하지만 사용자는 보통 한 번에 서비스 수행에 필요한 모든 정보를 제공하지 않고 지속적인 대화를 통해 정보를 제공하기 때문에 에이전트가 능동적으로 의도 파악에 참여하여 필요한 정보를 획득하는 기능이 필요하다. 실제 사람은 문제를 해결할 때에 문제에 대한 모든 정의와 조건을 기술한 후에 문제의 해결책을 찾는 경우는 거의 없으며, 오히려 문제에 대한 부분적인 정보를 교환해 가면서 문제를 점진적으로 구체화해간다[3]. 이런 대화의 특성을 기초로 한 상호주도형 대화 에이전트를 구현하기 위해서는 모든 상호작용이 문맥적으로 해석되어 사용자의 필요를 예측하고 사용자의 목적을 달성하기 위해서 가장 적절한 반응을 하도록 시스템을 구성해야 한다[3,4].

2.2 동적 베이지안 네트워크

동적 베이지안 네트워크(DBN; dynamic Bayesian networks)는 시간의 흐름에 따른 프로세스를 확률적으로 모델링하기 위해 시간상 변수들의 값을 바탕으로 설계한 베이지안 네트워크이다[5]. 기존의 정적인 베이지안 네트워크와 달리 특정 변수의 과거값이 현재 변수의 추론에 사용된다. 은닉 마르코프 모델이나 Kalman 필터 등도 간소화된 동적 베이지안 네트워크의 일종이며, 보통 음성인식 등과 같은 분야에 많이 적용되고 있다[6]. 그림 1은 동적 베이지안 네트워크의 예를 보여준다.

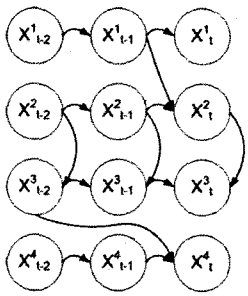

그림 1. 동적 베이지안 네트워크 예

동적 베이지안 네트워크는 각 시간 단위로 모델링하고자 하는 변수들로 구성된다. 보통 네트워크의 복잡도를 줄이기 위해서 다음 수식과 같이 마르코프 가정을 따라 m 길이의 과거값만을 사용하여 네트워크를 구성하기도 한다.

$$P(X^t|X^1, X^2, \ldots, X^{t-1}) \simeq P(X^t|X^{t-m}, X^{t-m+1}, \ldots, X^{t-1})$$

본 논문에서는 사용자의 일정에 대한 피로도를 추론하기 위해서 동적 베이지안 네트워크를 활용하였다.

2.3 ConceptNet

ConceptNet은 Liu 등이 개발한 상식 온톨로지로 모든 사물의 장소적, 물리적, 사회적 그리고 정신적인 양상을 표현하는 약 1.6 백만 개의 개념를 포함한다[7]. Musa 등은 GloBuddy 시스템에서 다양한 개념들로부터 중요한 개념을 추출하는 주제 추출 기능을 활용하여 각 상황에 맞는 외국어 문장들을 생성하였다[8].

본 연구에서는 ConceptNet의 guess_topic 함수를 사용한 주제 추출 기능을 활용하여 사용자의 의도와 사용자가 처한 환경에 맞는 서비스를 선택한다.

3. 제안하는 지능형 일정관리 에이전트

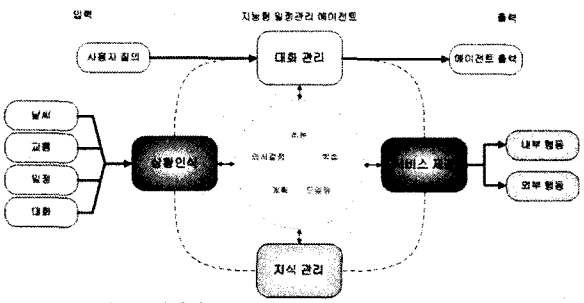

그림 2. 제안하는 지능형 일정관리 에이전트 구조

그림 2는 제안하는 지능형 일정관리 에이전트의 구조도를 보여준다. 대화 관리는 사용자 질의를 분석하여 사용자 의도를 파악한 후, 상호주도형 대화를 통해 사용자로부터 필요한 부가적인 정보를 추출한 후, 해당 서비스를 사용자에게 제공한다. 지식 관리는 본 논문에서 제안하는 상호주도형 대화를 위한 스크립트 언어로 작성된 스크립트와 사용자의 일정 정보를 관리하며, 상황인식에서는 날씨, 교통, 일정, 대화로그 등을 바탕으로 현재 상황을 인식하여 일정 추천에 이용된다.

3.1 상호주도형 대화를 위한 스크립트 언어

상호주도형 대화를 위해서 에이전트는 사용자 의도를 파악하고, 사용자로부터 추가로 필요한 정보를 스스로 획득해야 한다. 본 논문에서는 상호주도형 대화를 구현하기 위해 표 2와 같은 스크립트 언어와 표 3과 같은 내부 변수 표현을 설계한다.

표 2. 스크립트 언어

```
[topic_name] := @String
[var_string] := @String
[query_string] := @String
[answer_string] := @String
[function_name] := @String
[return_value] := @String

[topic] := <topic> [topic_name] </topic>
[pre_topic] := <pre_topic> [topic_name]+ </pre_topic>
[variable] := <variable> [var_string] </variable>
[necessary_var] :=
        <necessary_var> [variable]+ </necessary_var>
[candidate_var] :=
        <candidate_var> [variable]+ </candidate_var>
[query] := <query> [query_string] </query>
[internal_function] :=
          <change_topic> [topic_name] </change_topic>
        | <sub_topic> [topic_name] </sub_topic>
        | <end_topic/>
        | <extract_candidate_var/>
        | <answer> [answer_string] </answer>
[parameter] = [var_string] | &[var_string]
[function_form] := [function_name]({|[parameter]}+)
[return] :=
    <return value = [return_value]> [order]+ </return>
[function_body] := [function_form] [return]+
[extern_function] :=
                <function> [function_body] </function>
[order] := [internal_function] | [extern_function]
[action] := <action> [order]+ </action>
[topic_question_body] := [question_string] | [order]
[topic_question] := <topic_question>
                [topic_question_body] </topic_question>
[script_body] := [topic] ([pre_topic]) ([necessary_var])
([candidate_var]) [query]+ [action]+ [topic_question]+
[script] := <script> [script_body] </script>
```

표 3. 내부 변수 표현

```
[object_name] := @String
[var_string] := @String
[default_value] := @String
[value_string] := @String
[question_string] := @String

[object] := <object> [object_name] </object>
[variable] := <variable> [var_string] </variable>
[default] := <default> [default_value] </default>
[value] := <value> [value_string] </value>
[question] := <question> </question>
[tpl_question_body] := [question_string] | [order]
[tpl_question] :=
      <tpl_question> [tpl_question_body] </tpl_question>
[template_body] := [object] [variable] ([default])
                [value]+ [tpl_question]+
[template] := <template> [template_body] </template>
```

- 스크립트에서 대화의 상태를 나타내는 [topic]은 선택된 스크립트의 [topic]이 메모리 스택에 저장되어 대화의 흐름을 관리하고<end_topic/> 태그에서 메모리 스택에서 빠져나온다. 또한 스크립트 매칭과정에서 이전 주제에 대한 가중치를 부여하기 위해서 메모리 큐에도 저장된다. [pre_topic]은 주제의 변화를 원활하게 하기 위한 장치로, 메모리 큐에 [pre_topic]에서 명시

된 주제가 발견되면, 스크립트에 대한 가중치가 높아진다.

- [necessary_var]은 스크립트의 [action]이 수행되기 이전에 반드시 필요한 내부 변수를 명시하는 것으로 스크립트가 선택되었을 때, [necessary_var]에 대한 값이 존재하지 않으면 [necessary_var] 변수 값을 얻기 위해 에이전트가 사용자에게 질문을 하게 된다. 이때 사용할 질의문은 표 3의 내부 변수 표현 방법에 명시한다. [candidate_var]은 스크립트의 [action] 수행에 도움이 되는 변수로 [action]에서 반드시 필요로 하지는 않으며 <extract_candidate_var/> 태그를 만나면 [candidate_var]의 값을 추출하기 위한 질의를 수행한다.
- [query]는 스크립트 선택과 정보 추출에서 중요하게 사용되는 부분으로 사용자 입력 질의와 [query]에 명시되어 있는 문장과 패턴 매칭을 하여 높은 점수를 얻은 스크립트가 선택되게 된다. 스크립트 점수는 다른 주제에서의 동일한 사용자 입력을 처리하도록 [pre_topic] 변수 값을 활용하여 가중치를 부여한다.
- [action]은 선택된 스크립트를 실행할 준비가 되었을 경우([necess_var]을 모두 얻었을 경우) 실제 행동을 명시하는 부분으로 에이전트는 여러 [action]중 임으로 하나를 선택하고 그 안에 있는 [order]을 순서대로 수행하여 사용자에게 서비스를 제공한다. [order]는 [internal_function]과 [extern_function]으로 구분된다. [internal_function]은 시스템에서 미리 정의한 대화 수행에 필요한 명령어로 메모리의 변화와 에이전트 출력 문장을 명시한다. <change_topic>는 메모리 스택의 TOP에 위치한 주제를 변경하며, <sub_topic>은 메모리 스택에 새로운 주제를 푸쉬(push)한다. 해당 스크립트가 종료되었음을 알리는 <end_topic/>은 메모리 스택에서 해당 주제를 팝(pop)한다.
- <answer>는 [answer_string]을 답변으로 출력하며, [answer_string]에 내부 변수가 존재하면 해당 변수의 값을 출력한다.
- [extern_function]은 시스템에서 제공하지 않는 역할을 에이전트가 수행할 수 있도록 하는 사용자 정의 함수이다. 함수 원형은 리턴 값으로 string을 가지며, 파라메터로는 string 또는 string*로 제한한다. 사용자 지정함수의 수행결과에 따라 다른 행동을 수행할 수 있도록 [return]에 대해 정의한다. [topic_question]은 비슷한 점수를 가지는 복수의 스크립트가 선택되었을 때, 사용자의 의도를 보다 정확하게 파악하기 위해서 사용된다.
- 시스템 동작에서 사용될 값을 명시한 내부 변수는 [object]는 '일정,' '주소록' 등과 같은 개체의 범주에 대한 명시이고, [variable]은 추출할 변수를 그 추출 패턴과 함께 입력한다. [value]는 해당 변수가 가질 수 있는 값을 명시하며, 그 외에는 [default]를 사용한다. [tpl_question]은 해당 변수 값을 사용자에게 물어보기 위한 방법을 명시한다.

3.2 동적 베이지안 네트워크를 이용한 일정 피로도 추론

본 논문에서는 적절한 일정시간을 추천하기 위해서 동적 베이지안 네트워크를 이용하여 입력 일정에 대한 피로도를 추론한다. 먼저 사용자 프로파일로부터 사전에 입력된 일정 리스트를 호출하는데, 일정은 {대상(동료,친구,가족), 시간대(오전,오후,저녁), 일정유형(업무,약속,식사,무), 일정장소(내부,외부)}로 구성된다.

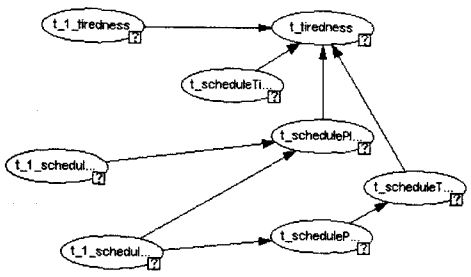

그림 3. 일정 피로도를 계산하기 위한 동적 베이지안 네트워크

t_scheduleType	ark										
t_schedulePlace	out										
t_scheduleTime_...	morning				afternoon				night		
t_1_tiredness	tired		vivid		tired		vivid		tired		vivid
tired	0.7		0.35		0.8		0.55		0.9		0.75
vivid	0.3		0.65		0.2		0.45		0.1		0.25

그림 4. 조건부 확률 테이블의 모습

새로 입력된 일정을 현재로부터 일주일 내의 적절한 시간대에 할당하기 위해 일주일 내의 비어있는 시간대 ET = {et_1, et_2, ..., et_m}를 추출하고 각 시간대에 신규 일정을 입력하였을 때의 피로도를 그림 3과 같은 구조의 동적 베이지안 네트워크를 이용하여 계산한다. 네트워크의 구조와 조건부 확률값은 일정 로그로부터 전문가가 설계하며, 그림 4는 조건부 확률값의 일부를 보여준다.

3.3 약속 장소 및 시간 추천

약속 장소 및 시간은 피로도 추론 모듈과 대화 모듈로부터 얻은 사용자의 의도와 환경정보를 사용하여 추천된다. 약속 장소의 추천을 위해 추천 가능한 장소들을 다음과 같이 모델링하였다.

P = { N, D, T, C }
N : 장소의 이름 (예: library of yonsei university),
D : 현재 위치에서 해당 장소까지의 거리(Km),
T : 교통의 좋고 나쁨(도보: 5, 환승 1회 : 4, 환승 2회: 3, 환승 3회 이상: 2, 시내: 1, 시외: 0),
C : 장소를 설명하는 개념 (예: library, park, farm 등)의 목록 = {c_1, c_2, ... , c_n}

약속 시간의 추천을 위해 다음과 같이 시간을 모델링하였다.

H = { $Period$, F, C }
$Period$: 두 개의 숫자의 쌍으로 표현된 시간대, 예:(12, 14)
F : 해당 시간대의 피로도 (0~100)
C : 시간대를 설명하는 개념 (예: morning, noon, night등)의 목록 = {c_1, c_2, ... , c_n}

약속 장소와 시간의 추천을 위해 우선 사용자의 환경 및 의도를 해당 개념으로 표현한다. 이 개념들을 ConceptNet의 guess_topic 함수에 입력하여 사용자의 상황에 맞는 주제목록 S = {($concept_1$, $score_1$), ($concept_2$, $score_2$), ... , ($concept_n$, $score_n$)}을 각 개념의 현재 사용자가 처한 상황에 대한 정확도와 함께 구한다. 이를 사용하여 각 장소 또는 각 시간대가 사용자의 상황과 얼마나 관련이 있는지를 평가하기 위해 장소와 시간대를 설명하는 각 개념들에 대해 다음과 같이 점수를 계산한다.

$$b_k = \begin{cases} score_m & c_k \in C,\ c_k = concept_m \\ 0 & c_k \notin C \end{cases}$$

각 장소들의 약속 장소로써의 적합도를 다음 식과 같이 구한다. 사용자가 일반적으로 피로도가 높을수록 가까운 거리를 선호하는 특성을 반영하고, 사용자 상황 및 약속 내용이 장소와 얼마나 가까운 관계를 갖는 지 ConceptNet을 통해 판단하였다.

$$f(P) = T \times (1 + \frac{(b_1 + b_2 + \cdots + b_n)}{n}) \times \frac{1}{(1+D)^{1+\frac{F}{100}}}$$

또한, 각 장소의 약속 시간에 대한 적합도는 다음 식을 통해 구한다. 피로도가 낮고, 사용자의 상황 및 약속 내용과 부합하는 시간대에 높은 적합도를 주었다.

$$f(H) = (1 + \frac{b_1 + b_2 + \cdots + b_n}{n}) \times (1 + \frac{100}{F})$$

최종적으로 적합도가 가장 큰 장소와 시간대를 사용자에게 우선적으로 추천한다.

4. 제안하는 일정관리 에이전트의 동작

본 논문에서는 그림 5와 같은 MFC기반의 일정관리 에이전트를 구현하여 제안하는 방법의 유용성을 확인하였다.

4.1 상호주도형 대화

표 4. 상호주도형 대화 예

1) 사용자: 오늘 식사 약속이 생겼어.
스크립트 선택: topic-일정추가
정보 추출: 일정종류-식사, 일정날짜-오늘
에이전트: 누구랑 만나기로 하셨나요?
정보 획득(질의): 일정대상
2) 사용자: 가족들이랑.
스크립트 선택: topic-일정추가
정보 추출: 일정대상-가족
에이전트: 신촌에서 보시나요?
정보 획득(추천): 일정장소
sub_topic: 일정추천
3) 사용자: 어
스크립트 선택: topic-일정장소추천
정보 추출: 일정장소-신촌
에이전트: 몇 시 약속인가요?
정보 획득(질의): 일정시간
4) 사용자: 1시에 보기로 했어.
스크립트 선택: topic-일정추가
정보 추출: 일정시간-1시
에이전트: 일정을 등록하였습니다.
서비스 제공: 일정등록

표 5. 내부 변수 표현 예

```
<template>
  <object> 일정 </object>
  <variable> $장소$ </variable>
  <value> 신촌 </value>
  <value> 서울역 </value>
  ...
  <tpl_question>
    <function> recommend_location|$일정$날짜$|$일정$시간
$|$일정$대상$|$일정$종류$
      <return value = false>
        <answer> 어디에서 모이나요? </answer>
      </return>
      <return value = true>
        <function> answer_recommend_location
                   |&$internal$ans$ </function>
        <answer> $internal$ans$ </answer>
        <sub_topic> 일정추천 </sub_topic>
      </return>
    </function>
  </tpl_question>
</template>
```

표 4는 상호주도형 대화의 예를 보여준다. 대화 1에서 에이전트는 사용자의 입력문과 스크립트의 [query] 간의 패턴매칭을 통해서 [topic]이 일정추가임을 알아내고, 필요한 정보를 추출한다. 해당 스크립트에서 [necessary_var]이 모두 추출되었는지 확인하고, 대화 1에서와 같이 [necessary_var]이 모두 존재하지 않으므로 [necessary_var] 변수 중 하나를 선택하여 내부 변수 표현의 [tpl_question]을 수행한다. [tpl_question]은 문장을 통해서 사용자에게 질의를 던지거나 대화 2와 같이 추천를 수행하기도 한다 (표 5 참고). 대화 4에서는 일정등록에 필요한 모든 변수가 추출되어 실제 서비스가 동작한다. 표 5는 표 4의 대화에서 사용된 내부 변수 표현을, 표 6은 표 4의 대화를 위한 스크립트의 일부를 보여준다.

표 6. 스크립트 예

```
<script>
  <topic>일정입력</topic>
  <necessary_var>
    <variable> $일정$날짜$ </variable>
    <variable> $일정$시간$ </variable>
    <variable> $일정$유형$ </variable>
    <variable> $일정$대상$ </variable>
    <variable> $일정$장소$ </variable>
  </necessary_var>
  <query>$일정$날짜$ $일정$종류$가에 생겼어. </query>
  <query>$일정$대상$이랑. </query>
  <query>$일정$시간$에 보기로 했어. </query>
  ...
  <action>
    <function> SchedInput|$일정$날짜$|$일정$시간$|$일정
$대상$|$일정$종류$|$일정$장소$
      <return value=true>
        <answer> 일정을 등록하였습니다. </answer>
        <end_script/>
      </return>
      <return value=conflict>
        <function>
          QuesSchedInputConflict|&$internal$answer$
        </function>
        <answer> $internal$answer$ </answer>
        <change_topic> 일정충돌 </change_topic>
      </return>
    </function>
  </action>
  <topic_question> 일정입력하시게요? </topic_question>
</script>
```

4.2 피로도 예측 동작 시나리오

표 7. 삽입된 일정 리스트

일정 ID	대상	시간대	일정유형	일정장소
1	친구	오전 (10시)	약속	외부
2	가족	오후 (12시)	식사	외부
3	동료A	오후 (15시)	업무	내부
4	동료B	오후 (17시)	업무	내부
5	동료C	저녁 (19시)	업무	외부

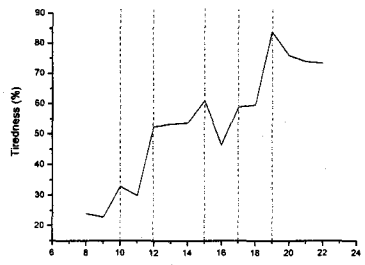

그림 5. 피로도 예측 결과

표 7과 같이 구성된 일정 리스트에 대해 동적 베이지안 네트워크를 이용하여 그림 5와 같이 당일 피로도를 예측하였고, 표 8과 같이 구성된 일정 리스트에 대해서는 그림 6과 같은 피도로 예측 값을 획득하였다.

표 8. 삽입된 일정 리스트

일정 ID	대상	시간대	일정유형	일정장소
1	동료A	오전 (8시)	업무	내부
2	친구	오후 (13시)	약속	내부
3	동료B	오후 (18시)	업무	외부

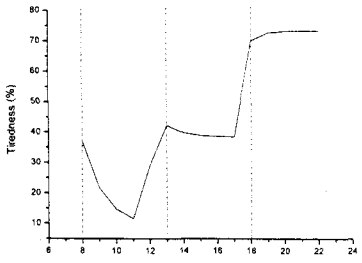

그림 6. 피로도 예측 결과

표 8이나 표 10에서와 같이 일정을 수행한 후에 피로도가 높아졌으며, 특히 외부에서 일정을 수행하였을 경우 피로도가 급격이 높아졌다. 일정이 없는 시간대에는 피로도가 점차 감소하는 것을 확인하였으며, 저녁 보다는 오전에 사용자 피로도가 낮았다. 내부에서 일정이 있을 경우에는 일정 이후에 피로도가 빠르게 감소하였지만, 외부에서 일정을 수행한 후에는 외출 중이기 때문에 피로도가 천천히 감소하였다.

4.3 약속 장소 및 시간 추천

제안하는 방법의 유용성을 확인하기 위해, 사용자가 약속을 잡는 대표적인 상황을 바탕으로 다음과 같이 장소와 시간을 추천하였다. 장소의 경우 표 9의 장소들에 대해 추천을 시행한 결과, 표 10과 같이 피로도가 높을수록 가까운 곳의 적합도가 상승하고, 상황에 맞는 경우 적합도가 상승함을 확인하였다. 시간대의 경우 표 11의 시간들에 추천을 시행한 결과, 표 12와 같이 피로도가 낮고, 상황에 맞는 시간대가 높은 적합도를 얻은 것을 확인하였다.

표 9. 추천에 사용된 장소 목록

ID	장소명(N)	거리(D)	교통(T)	설명(C)
P1	중앙도서관	0.283	5	library, university
P2	공학대학	0	5	engineering, university
P3	서울역	3.244	4	train, station

표 10. 각 장소별 적합도

(상황, F)	(travel,50)	(study,50)	(study,70)
P1	3.44	7.17	6.82
P2	5.0	7.80	7.80
P3	0.80	0.46	0.34

표 11. 추천에 사용된 시간대 목록

ID	시간대(Period)	피로도(F)	설명(C)
H1	9 ~ 10	21.7	morning
H2	13 ~ 14	42.3	noon, afternoon
H3	20 ~ 21	73.2	afternoon, evening

표 12. 각 시간대별 적합도

상황	study	lunch
H1	5.61	5.61
H2	3.36	4.58
H3	2.37	2.796

5. 결론

본 논문에서는 상호주도형 대화와 상황인식을 통한 지능형 일정관리 에이전트를 제안하였다. 제안한 방법에서는 상호주도형 대화를 위한 스크립트 언어를 통해 사용자와 에이전트 간의 원활한 대화를 제공하고, 동적 베이지안 네트워크와 ConceptNet을 활용한 상황인식 서비스를 제공하였다. 다양한 시나리오를 통해 제안하는 지능형 에이전트의 유용성을 확인하였다.

감사의 글

이 연구는 산업자원부가 지원한 뇌과학 연구 프로그램에 의해 지원되었음

참고문헌

[1] S. Schiaffino and A. Amandi, "On the design of a software secretary," *In Proc. Of the Argentine Symposium on AI*, pp. 218-230, 2002.

[2] S. Bocionek, "Software secretaries: learning and negotiating personal assistants for the daily office work." *In Proc. IEEE Int. Conf. on Systems, Man and Cybernetics*, pp. 7-12, 1994.

[3] J. Allen, C. I. Guinn and E. Horviz, "Mixed Initiative Interaction," *IEEE Intelligent Systems*, vol. 14, no. 5, pp.14-23, 1999.

[4] E. Horvitz, J. Breese, D. Heckerman, D. Hovel and K. Rommelse, "The lumiere project: bayesian user modeling for inferring the goals and needs of software users," *Proc. of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, p.256-265, 1998.

[5] X. Li, et al., "Active affective state detection and user assistance with dynamic Bayesian networks," *IEEE Trans. Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 35, no. 1, pp. 93-105, 2005.

[6] A. Tucker, et al., "Evolutionary learning of dynamic probabilistic models with large time lags," *Int. J. Intelligent Systems*, vol. 16, no. 5, pp.621-646, 2001.

[7] H. Liu, P. Singh, "ConceptNet: A practical commonsense reasoning toolkit," *BT Technology J.*, vol. 22, 2004.

[8] R. Musa, M. Scheidegger, A. Kulas and Y. Anguilet, "GloBuddy, a dynamic broad context phrase book," *Proc. Of Context 2003*, pp 467-474, 2003.

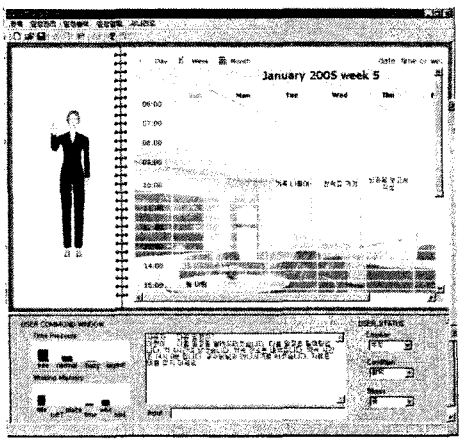

그림 5. 일정관리 에이전트 구현모습