제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

분산 제약조건 만족 특성을 이용한 다중 에이전트 기반 강의시간표 작성 시스템

황경순*, 전중남*, 이건명*
*충북대학교 컴퓨터과학과
첨단정보기술 연구센터
hks@aicore.chungbuk.ac.kr

A Multiagent-based Lecture Timetabling System using the Properties of Distributed Constraint Satisfaction

Kyoung-Soon Hwang*, Joong-Nam Jun*, Keon-Myung Lee*
*Dept. of Computer Science, Chungbuk National University and AITrc

요 약

본 논문에서는 대표적인 NP 문제의 하나인 강의시간표 문제를 분산 제약조건 만족 문제로 해결하는 시스템을 제안한다. 제안한 시스템에서는 복잡하고 방대한 강의시간표 문제를 여러 개의 작은 모듈 단위의 에이전트로 분할한 후 개별 문제의 해를 구하고 이들을 결합시켜 가능해(feasible solution)를 찾는다. 한편, 분리된 에이전트에 의해 해결되는 부-문제들이 상호 의존적이면서 중첩된 경우에 해들 사이의 모순을 해결하면서 최종 해를 구한다. 제안한 시스템에서는 다음과 같은 방법으로 문제를 해결한다. 제약조건을 점진적으로 추가하여 탐색공간을 줄여 나간 후, 국소 탐색을 통해 변수에 일관된 도메인 값을 할당한다. nogood에 대하여 점진적인 제약조건 완화로 탐색공간을 확장하여 모든 변수에 도메인 값을 배정한다. 제약조건 완화는 제약조건들을 몇몇 단계로 정의하고, 휴리스틱 순서와 제약조건의 중요도에 따라 되추적 탐색 기법을 이용하여 순차적으로 완화한다. 만일 과잉-제약조건이 발생할 경우 가중치의 합이 최소로 하는 값을 배정한다. 즉 모든 변수에 도메인 값은 모든 제약조건을 만족하는 초기의 부-문제에 가능해가 될 수 있는 제약조건을 만족해야 한다.

1. 서 론[1)]

강의시간표 문제는 일주일을 기본으로 하는 특정 시간대 별로 학생, 교수, 그리고 강의실과 같은 자원에 대한 스케줄링 문제이며, 또한 각각의 자원에 대하여 적절한 조합으로 제약조건들 사이의 충돌을 최소화하여 자원을 배치하는 할당문제이다.

강의시간표를 작성하기 위한 연구는 많이 수행되어 왔다. 그래프 컬러링(Graph Coloring)[1], 수학적 프로그래밍(Mathematical Programming)[2,3], 타부탐색(Tabu search)[4], 최소충돌(Minimizing Conflicts)[6], 그리고 유전자 알고리즘(Genetic Algorithm)[5,7]등과 같은 방법들이 있다. 대규모의 제약조건 만족 문제와 스케줄 문제를 해결하기 위한 휴리스틱 탐색 전략으로는 Forward Checking, Backtracking, Backjumping, Variable Ordering, 그리고 Preprocessing CSPs등을 사용해 왔다.[10] 제안된 시스템은 자원 공유를 통해 각각의 자원들 사이에 중복이 없이 교수와 학생이 원하는 시간표를 작성하는 것이다. 또한 강의시간표 작성 후 강의시간 변경이 제공되는 정보를 통해 손쉽게 이루어질 수 있을 뿐만 아니라 각종 보고서 및 통계자료를 생성할 수 있다. [그림 1]은 여러 개의 에이전트로 구성된 다중 에이전트 시스템 구성도이다. 분산된 자원과 제약조건들에 대하여 모듈 단위의 에이전트별로 개별 강의시간표를 작성한 후 저장된 정보를 공유하여 각각의 에이전트 사이에 모순된 해를 해결한다.

이 논문은 첨단 정보기술 연구센터(AITrc)를 통해서 과학재단의 지원을 받은 것임.

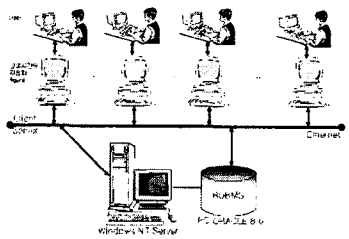

그림 1 다중 에이전트 구성도

본 논문의 구성은 다음과 같다. 2절에서는 강의시간표 문제를 설명하고, 3절에는 구현을 기술하고, 마지막 4절에서는 본 논문의 결론과 향후 과제에 대하여 논한다.

2 강의시간표 문제 (Lecture Timetabling Problem)

2.1 자료표현 (Data representation)

모든 학생들은 수강신청 과목에 관한 강좌에 출석해야 한다, 수업유형은 강의, 세미나, 그리고 실습과 같은 형태들로 구성된다. 일반적으로 강의(이론)에 대해서는 40/80/120명 단위의 그룹으로 수업을 한다. (실습인 경우 40명 단위의 그룹). 반은 합반(0), A반, B반, ...으로 구분한다. 강의시간은 50분을 기준으로 하고 시수만큼 계산된다 (이론은 1시간을 1학점으로 하고 실습은 2시간을 1학점으로 시수를 계산한다). 강의시간표에 대한 해는 S={학과,학년, 반, 과목, 교수, 강의 시작시간, 강의 종료시간, 강의실}로 한다. 강의시간표에 대한 모델은 아래와 같다.[11]

Day = {Mon, Tue, Wed, Thu, Fir, Sat}
Time_slot = {1교시, 2교시, 3교시,… }
Room = {A101, A102, ..., B102, B102, ...}
Professor = {ID1, ID2, ...}
Course = {Course1, Course2, ...}
…

이것은 요일(Day), 시간(Time_slot), 강의실(Room), 교수(Professor), 과목(Course)등 각각의 값에 집합으로 이루어져있다. 수학적 프로그래밍 모델로 형식화(formulating)한다면 다섯 개의 튜플(tuple:속성)을 변수 $X_{a,b,c,d,e}$로 나타낼 수 있다. 튜플들은 a=요일, b=시간, c=강의실, d=교수 그리고 e=과목이다. 각 변수에 대한 튜플들을 정수로 표현하여 변수들 사이에 일-대-일 대응(mapping)을 얻는다. 예를 들어, 다음과 같이 $X_{1,2,101,200177,101015}$을 생성한다. 즉 1(월요일), 2(2교시), 101(강의실101호), 2(2교시), 200177(김영희), 101015(인터넷 활용)과 같이 표현된다.

2.2 문제해결 방법(Solution method)

2.2.1 제약조건 만족 문제 (CSP)

제약조건 만족 문제의 정의는 다음과 같다. 변수들 $X=\{x_1,\ldots,x_n\}$ n의 집합, 각각의 x_i 에 대하여 도메인 D_i 값의 집합, 그리고 제약조건들의 집합이다. 각각의 도메인 값들의 유한한 집합이며, 그 중에 하나는 대응하는 변수에 할당되어야 한다. 변수가 도메인부터 값을 할당받았을 때 활성화되었다고 하고 그렇지 않으면 비-활성화되었다고 한다. 제약조건의 만족문제의 해는 모든 제약조건들이 만족하는 변수들의 값을 찾는 것이다.[9]

2.2.2 분산 제약조건 만족 문제 (DCSP)

분산제약조건 만족 문제는 제약조건 만족 문제와 동일하게 정의된다.[9]

① 분산제약조건 만족 문제는 에이전트간에 sending message에 의해 통신을 한다.
② 메시지 지연은 유한하다.
③ 분산제약조건 만족 문제는 n개의 에이전트 $1,2,3,\ldots,n$가 존재하고 각각의 에이전트에는 변수가 존재한다. 에이전트 i는 연관된 모든 변수의 제약조건을 알고 있다. R_i로 제약조건을 표현하고, 에이전트 i는 다른 에이전트들의 관련된 식별자(각 변수는 인수로 R_i을 포함)를 알고 있다.

2.2.3 제약조건 모델링 (Constraint Modeling)

제약조건 기반의 문제해결은 인공지능 분야에서 많은 관심을 갖는 것 중하나이다. 그 이유는 문제 해결에 위한 강력한 방법들을 제공되기 때문이다. 즉 계획, 자원할당, 그리고 스케줄링과 같은 많은 문제들이 제약조건 모델링이 가능하다. 또한 제약조건 모델링 방법에 따라 해 탐색에 있었어 큰 영향을 가진다. 제약 조건을 두 종류로 분류하면, 우선 반드시 지켜져야 되는 강한-제약조건(Strong Constraint)과 약간의 유연성을 가진 약한-제약조건(Weak Constraint)이다[11].

강한-제약조건(SC)
C1:한 강의실에서는 동시에 두 강의를 할 수 없다.(0)
C2:한 교수가 동시에 여러 강의실에서 강의를 할 수 없다.(예외 - 교수가 카메라와 비디오 프로젝션(영상강의), 그리고 컴퓨터(Internet 강의)의 도움을 받아 동시에 여러 강의실에서 한 과목에 대하여 강의할 수 있다)(0)
C3:특정 과목들은 겹치지 않는다.(0)
C4:시간 배정형식을 지킨다.(0)
C5:각각의 강의실은 학생들의 수에 기대되는 충분한 공간을 가지고 있어야 한다.(0)
C6:각각의 강의실은 강사의 요구에 대하여 적절한 장비를 갖추고 있어야 한다.(0)
C7:교수별 하루에 최대 수업 시간표를 제한한다. (0)

약한-제약조건(WC)
C8:교수의 제한시간을 보장한다.(1)
C9:교수의 선호시간을 보장한다.(2)
C10:강의실 배정은 학생의 최소한 움직임을 보장한다 (2)
C11:학생들의 식사시간을 보장한다. (2)
C12:너무 이른 아침이나 너무 늦은 저녁은 제한한다. (3)

위의 제약조건들에 대하여 누적 제약조건으로는 $cumulative([S1,\ldots,Sn],[D1,\ldots,Dn],[R1,\ldots,Rn],L)$로 한다. 여기서 Si는 태스크의 시작을 표현한 것이며 Di는 기간 그리고 Ri는 각각의 태스크에서 필요로 하는 자원의 총계이다. L은 각각의 시간에 대한 사용 가능한 자원의 총계이다. 누적 스케줄링은 시간 i의 스케줄에 대한 동안에 자원에 대한 총 사용은 L을 초과할 수 없다.[8] 변수들의 튜플은 a=(요)일, b=시간, c=강의실, d=교수 그리고 e=과목이다.

- C4, C11, C12는 직접 도메인을 제한한다.
- C1은 누적제약조건에 따라
  $\sum_{d\in D_b}\sum_{c\in C_b}[X_{a,b,c,d,e}]\le 1$, a∈A, b∈B
  D_b는 교수(d)에 강의실(C)이 배정되었을 때, 교수(d)에 의해 강의실(c)에서 e(과목)의 수업시간을 배정
- C2, C3는 C1과 동일하다
- C7는 C1과 같이 누적 제약조건 적용
  $\sum_{d\in D_b}\sum_{a\in A_b}[X_{a,b,c,d,e}]\le 8$, a∈A, b∈B
  교수가 하루에 8시간을 초과해 강의할 수 없다.

강한 제약조건은 위배할 수 없는 제약조건으로 주로 약한 제약조건을 완화하는데 각각의 대학마다 특성이 있기 때문에 정형화한다는 것은 거의 불가능하다. 단지 해를 탐색하는데 탐색공간을 줄이기 위하여 사용된다.

2.2.4 제약조건 완화(Constraint Relaxation)

제약조건을 완화하기 위해서는 변수 중요도를 순서화(Variable Ordering)하므로 작업흐름에 따라 해의 탐색에 큰 영향을 미친다. 제약조건을 계층화하여 각각에 제약조건에 가중치를 할당하는데 이때 가중치는 제약조건의 중요도와 관계가 있다. 즉, 가중치가 클수록 제약조건의 중요도가 적게 된다. 제약조건 완화는 가중치가 큰 순서로 완화하고 만일 과잉-제약조건이 발생할 경우 가중치의 합이 최소로 하는 값을 배정한다. 즉, 모든 변수에 도메인 값은 모든 제약조건을 만족하든 초기의 부분 문제에 가능해가 될 수 있는 제약조건을 만족해야한다.

2.3 강의시간표 알고리즘

대부분의 제약조건 만족 문제는 되추적 알고리즘이 기본이 된다. 이 알고리즘은 시스템적(systematic)탐색을 하고, 해가 존재한다면 그 해를 찾는 것을 보장한다. 강의시간표 문제에선 이 탐색 기법은 탐색 공간이 기하급수로 증가하는 NP-Complete이다. 하지만 제약조건 완화기법과 탐색 공간 제한(pruning)기법에 의해 개선 될 수 있다. 그 외에도 국소 탐색(local search)있는데, 이 방법은 탐색 공간에 대하여 가능성에 근거한 전개를 수행한다. 그러므로 반드시 해를 찾는 다는 보장은 할 수 없다. 본 논문에서는 이 두 가지 방법을 적절히 혼합하여 사용한다. 1) 제약조건을 점진적으로 추가하여 탐색공간을 줄여 나간(domain filtering)후 국소 탐색(local search)을 통해 변수에 도메인 값을 할당한다. 2) nogood에 대하여 점진적으로 제약조건 완화하여 탐색공간을 확장하여 모든 변수에 값을 배정한

다. 제약조건 완화는 제약조건들을 몇몇 단계로 정의하고, 휴리스틱 순서와 제약조건의 중요도에 따라 되추적 탐색 기법을 이용하여 순차적으로 완화한다. 전체 작업의 흐름은 다음과 같다.

① 교양과목 전처리
② 각 학과 학년/반별 개설강좌 생성
 -교수에 정보공유 (한 교수가 두개 이상의 학과에 강의하는 경우)
 -강의실 정보공유(타 학과, 공용, 그리고 학부 강의실 사용하는 경우- 학년별 이론과 실습 과목을 분리하여 [그림 2]와 같이 강의시간 배정형식을 적용 (학년별 교차 배정)
③ 전체 통합 조정
⑤ 강의시간표 생성

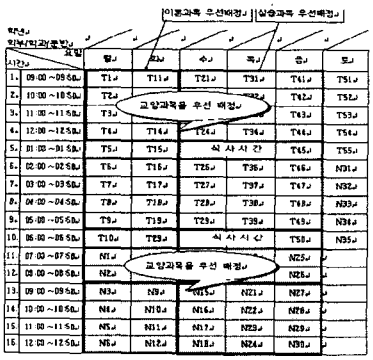

그림 2. 강의시간 배정형식

2.3.1 강의시간 배정 알고리즘

초기 강의시간 배정은 Forward 단계를 걸쳐 강의시간 배정을 한 후, nogood를 형상화한다. [그림3]은 강의시간 배정 프로시저이다. 또한 nogood는 제약조건 위배를 수정하는 동안 단순한 휴리스틱 순서에 의해 유도할 수 있으며, 미해결 제약조건 위배의 개수를 최소화하는 새로운 값을 선택한다. 이 과정은 다음과 같다.

① nogood 자료 생성
② 경험적 방법에 의해 변수 순서화
③ 제약조건만족 문제 해결
 - 연강시간을 최대로 하다.
 - 공강시간을 최소로 한다
 - 과잉-제약조건이 발생할 경우

$$F(c) = Min\left\{ \sum_{i=1}^{n} \sum_{j=1}^{n} W_{i,j} \right\},\ i=1,2,\cdots,n\ \text{(variable)},\ j=1,2,\cdots,n\ \text{(Domain)}$$

변점에 대한 가중치(W)의 합을 최소화한다.
④ 변수 활성화(Instantiated value of variables)
⑤ 만약 변수 활성화에 실패하면 ③~④ 반복

2.6.2 강의실 배정 알고리즘

만약 강의실 배정을 임의의 학과 순서로 배정했을 경우 나중에 배정이 되는 학과는 반드시 강의실이 부족하다. 하지만 학부/학과에 전용 강의실을 지정하여 강의실 사용에 우선권을 부여한다면 위의 문제를 다소 해결 할 수 있다. 강의실 배정은 이론 과목은 수강인원을 고려하여 우선 학과 전용 강의실을 탐색하여 배정한 후 강의실의 중복되는 경우 공동 강의실, 학부 그리고 동일 건물 순서로 탐색하여 일괄 배정이 가능하다. 실습 과목은 과목별 실습실 종류가 다양하므로 우선, 과목변수에 주석 처리하여 용도에 맞는 강의실을 배정한다. [그림4]는 강의실 배정 프로시저이다

```
Variable : X = {year, hakgi, hakkwa, kwamok, koosu, ban,
                is_gb, l_ty, l_times, hakyun, , p_time, r_time}
Domain   : D = {T1, T2, ..., T55, N1, N2, ..., N35}

Procedure Timetabling
Input : Variables = {X1, X2, ...Xn}, Domains = {D1, D2, ..., Dn}
Output : Either Solution or Nogood

Initialization
Heuristically order varibles
DO WHILE X is NOT NULL
  i++
  X_i ← FetchVariable
GenerateCSP(thresholds)     // Heuristically Method -> threshold 결정

IF 선호시간이 존재한다면
   IF 선호시간이면 해당 시간이 배정되어 있지 아니면
      {강의시간 수만큼 연속 배정}
       ins_chk = 1
   END IF
ELSE
   IF 해당 시간이 배정되어 있지 않고 제한시간이 아니면
      {강의시간 수만큼 연속 배정}
       ins_chk = 1
   END IF
END IF

IF ins_chk = 1 THEN
   Instantiated value of  X_i
ELSE
   nogood of  X_i                  // inconsistent variable
END IF
LOOP
```

그림 3. 강의시간 배정 알고리즘

```
Procedure RoomSet
Input : Variables = {X1, X2, ...Xn},
        R = {build, room, g_type, inwon, g_hakkwa, g_hakbu}
Output : Either Solution or Nogood

Initialization
Heuristically order varibles
DO WHILE X is NOT NULL
  i++
  X_i ← FetchVariable
IF 수업유형이 이론이면
    FOR j = 1 TO n STEP 1
        {지정학과 , 학부, 그리고 동일 건물 순서로 강의실을 검색}
        {강의실배정}
         ins_chk = 1
    NEXT
ELSE                                        //실습
    FOR j = 1 TO n STEP 1
        {과목별 강의실 유형 선택}
         {지정학과 , 학부, 그리고 동일 건물 순서로 강의실을 검색}
         {강의실 배정}
          ins_chk = 1
    NEXT
END IF

IF ins_chk = 1 THEN
   Instantiated value of  X_i
ELSE
   nogood of  X_i                  // inconsistent variable
END IF
```

그림 4. 강의실 배정 알고리즘

3.구현

본 논문에서 제안한 시스템은 충청대학을 대상으로 windows 2000에서 Powerbuilder 7.0과 PC ORACLE 8.0 등을 사용하여 구현하였다. [그림 5]는 시스템의 login 화면으로 ID를 이용하여 사용자별 시스템 사용 권한 제한한다. 예를 들어 관리자는 시스템 전체, 교수는 스케줄 관리와 리포트 결과 그리고 학과에서는 학과 강좌개설과 리포트 결과를 사용 할 수 있다.

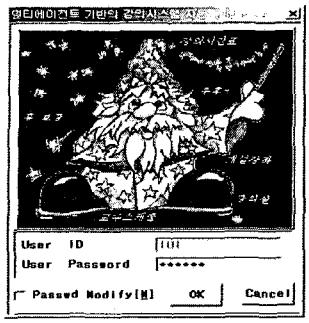

그림 5. Login 화면

강의시간표 작성을 위하여 강의시간 배정에 관한 변수, 도메인을 생성한다. (변수={학년도, 학기, 학과, 이수구분, 수업유형, 수업시간, 교수, 과목, 반, 주야구분, ...},도메인={T1, T2, ..., T55, N1, N2, ..., N35}) 그리고 교양과목, 학과별 개설강좌 그리고 통합 조정 후 강의시간표를 생성한다. [그림 6]은 교양 과목 강의시간 배정 결과이다 (수업담당자가 전체 학과를 일괄처리) [그림 7]은 학과별 강의시간 배정 결과이다. (교양과목과 교수의 선호/제한시간을 기본으로 학과, 학년 그리고 반별로 강의시간을 배정) [그림 8]은 강의실 배정 결과이다.

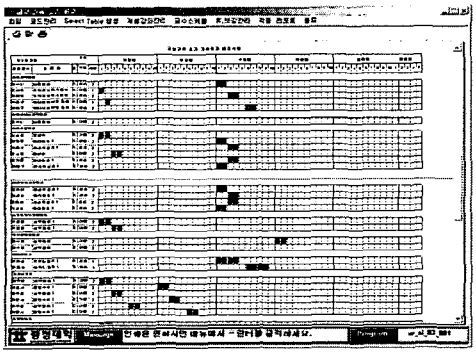

그림 6 교양과목 강의시간 배정 결과

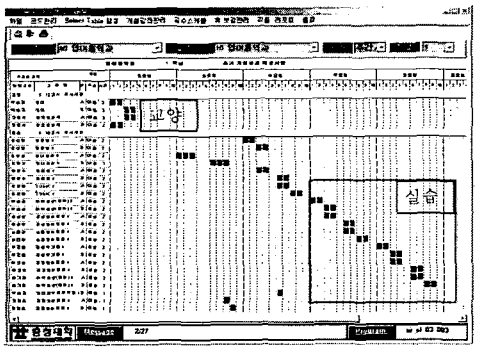

그림 7. 전공 과목 강의시간 배정 결과

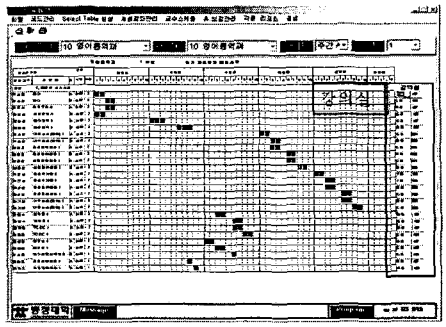

그림8. 강의실 배정 결과

4. 결론과 향후 과제

본 논문에서는 강의시간표를 작성하기 위하여 여러 개의 작은 모듈 단위의 에이전트로 분할하여 개별 해를 구한 후 그 해를 결합하여 최종 해 구하는 방법으로 분산 제약조건 만족 특성을 이용한 멀티에이전트 기반의 강의시간표 자동화 시스템을 제안하였다. 하지만 시스템 구현에 있어 제약조건을 코딩하는 과정이 너무 많고 복잡하여 유지 보수에 문제점이 있다. 향후 과제로는 강의시간표에 대한 제약조건에 규칙을 자동 생성하여 제약조건을 프로그래밍이 가능한 시스템에 대한 연구가 필요하다.

참고문헌

[1] Werra D., Chalhal, "An interactive System for Constructing Timeables on a PC", European Journal Operation Research 40 (1989)
[2] Akkoyunlu EA, "A Linear algorithm for computing the optimum university timetabling", Computer J 16 (1973)
[3] Lawrie N.L., "An Integer Programming Model of a School Timetabling problem", Computer Journal 12(1969)
[4] Hertz, A., "Tabu Search for Large scale Timetabling Problem", European Journal of Operation Research 54(1991)
[5] Hsiao-Lan Fang, "Genetic Algorithm in Timetabling and Scheduling", 1994
[6] Steven Minton Andy Philips, Mark D . Johnston, Philip Laird, "Minimizing Conflicts : A Heuristic Repair Method for Constraint-Satisfaction and Scheduling Problems".
[7] Wilhelm Erben, Jurgen Keppler, "A Genetic Algorithm Solving a Weekly Course-Timetabling Problem", Proceedings of the First International Conference on the Practice and Theory of Automated Timetabling
[8] Hana Rudov and Ludek Matyska., "Constraint-based timetabling with student schedules". In Edmund Burke and Wilhelm Erben, editors, PATAT 2000 - Proceedings of the 3rd international conference on the Practice And Theory of Automated Timetabling
[9] Makoto Yokoo, Toru Ishida, Edmund H. Durfee, Kazuhiro Kuwabara, "Distributed Constraint Satisfaction for Formalizing Distributed Problem Solving", Proc. of the 12th International Conference on Distributed Computing Systems (ICDCS-92)
[10] Rina Dechter,"Backtracking algorithms for constraint satisfaction problems", Department of Computer and Information Science University of California, 29, 1997
[11] 황경순, 진중남, 이건명, "분산 제약조건 만족 특성을 이용한 멀티에이전트기반 강의 시간표 자동화 시스템 설계", 한국정보과학회 '2002 봄 학술 발표 논문집(B),29권 1호