제약식프로그래밍과 최적화를 이용한 하이브리드 솔버의 구현

On Implementing a Hybrid Solver from Constraint Programming and Optimization

김 학 진 (Hak-Jin Kim)　　연세대학교 경영대학 조교수

요　약

제약식 프로그래밍과 최적화 솔버는 공통된 문제를 풀기 위한 해법으로서 서로 다른 영역에서 발전되어왔다. 특히 제약식 확산법과 선형 계획법은 두 영역의 주된 기법으로서 조합 최적화 문제를 푸는데 함께 사용될 수 있는 통합 가능한 보완 기법들이다. 지금까지 이를 통합하기 위한 시도는 주로 한 기법을 다른 기법의 모형 틀안에 포함시키는 것이었다. 본 논문은 둘의 통합을 통한 잇점들은 충분히 사용하기 위해서는 모형 역시 통합될 필요가 있음과 그 모형 통합의 틀을 보이고 그 틀 안에서 어떻게 두 기법의 솔버의 수준으로 통합되어 새로운 혼합 솔버를 구축할 수 있는지를 보인다.

키워드 : 제약식 프로그래밍, 제약식만족 문제, 조합 최적화 문제, 제약식 확산법, 선형 계획법, 정수 계획법, 소프트웨어

Ⅰ. 서 론

최적화 특히 조합 최적화 문제는 산업 현장에서 벌어지는 다수의 의사결정 문제를 해결하기 위한 해법으로 가장 먼저 생각되는 의사결정 기법들 중의 하나이다. 컴퓨터와 통신의 발달은 지금까지 산업 생산계획에서 나올 수 있는 의사 결정 문제가 동일하게 통신의 문제로 확장됨을 보여준다. 다양한 네트워크 상에서의 스케줄 문제와 설계의 문제, 인터넷의 발전과 더불어 해결해야할 통신자원 배분과 가격결정 등 경제적인 문제들, 그리고 인터넷이라는 새로운 맥락에서 벌어지는 모든 온라인 산업 경제의 문제들은 벌어지는 배경은 다르지만 전통적인 오프라인 산업의 문제들과 구조적으로 동일하다. 특히 조합 최적화 문제는 OR분과에서 오랫동안 효율적인 해법을 찾기 위해 노력해온 문제이다. 본 논문은 OR분야와 다른 배경을 가진, 최근에 전산학의 인공지능 분과에서 조합 최적화 문제를 풀기위한 방법으로 각광을 받기 시작한 제약식 프로그래밍의 장점을 OR의 최적화 기법과 결합하여 시너지 효과를 올릴 수 있는 틀을 생각해보고 실제 솔버를 구현할 때 고려해야할 점들을 제안하고자 한다.

정수 계획법(Integer Programming, IP)과 제약식 프로그래밍(Constraint Programming, CP)은 단체법(Simplex Method)라는 선형계획(Linear Programming, LP) 풀이 기법과 제약식 확산법(Constraint Propagation)

이라는 두 개의 서로 다른 문제 풀이 기법을 가지고 있다. 이 둘은 응용분야에서 많이 나타나는 조합 최적화(Combinatorial Optimization) 문제를 풀기 위해 서로의 장점을 사용할 수 있는 통합의 잠재성을 지닌 기법들이다. 하지만 그런 통합의 노력은 최근에서야 나타나고 있다. IP는 자본예산(Capital budgeting), 빈패킹 문제(Bin Packing), 승무원 스케쥴링(Crew Scheduling), 외판원 경로문제(Traveling Salesman Problem)와 같이 다양한 조합 최적화 문제에 지금까지 성공적으로 적용되어온 방법이다. CP는 최근 10년간 스케쥴링, 계획과 할당 등과 같이 순열, 이산화, 대칭 등의 구조를 포함하는 조합 최적화 문제에 적용되어 유연하며 효율적인 방법으로 알려진 상업적으로 성공한 방법이다. 이 둘은 공통적으로 같은 영역의 문제에 관심을 기울임으로써 문제를 좀더 효율적으로 풀 수 있는 방법을 모색하게 되었고 자연스럽게 통합의 가능성을 던지게 되었다.

CP나 IP 모두 기본적인 방법으로서 탐색 공간의 해를 열거하는 가지치기에 의존한다. 가지치기의 틀 안에서 CP나 IP는 문제 풀이에 추론(Inference)과 탐색(Search)이라는 쌍대적인 접근 방법이다. CP는 이 중 제약식 확산법이라는 형태로 추론을 더 강조한다. 즉 변수의 도메인에서 실현 불가능한 값들을 제거해 나가는 것이다. 이것은 IP와 같이 문제의 해를 찾기까지 연속의 완성된 해(CP에서는 완전표기 (Complete Labelling)라 부른다.)를 찾아 나가는 탐색과는 차이가 있다. IP는 선형계획 모형으로 표현되는 이완식(Relaxation)을 탐색나무의 각 노드에서 풀어 완성된 해를 찾고 탐색을 통해 이를 계속 개선해 나간다.

IP는 이완식을 더욱 조여주기 위해 주어진 제약식들로부터 선형 부등식을 도출하는 절면 추출(Cutting Plane Generation)법에서 큰 잇점을 지니는데, 특히 문제가 다면체 분석(Polyhedral Analysis)을 수행하기 쉬울 때 강력한 도구로서 사용된다. 그러나 IP는 그 제약식이 정수 변수를 포함하는 부등식이나 등식의 꼴로 표현되어야 한다는 언어적인 단점이 있다. 만약 그렇게 표현되지 않으면 앞에 말한 훌륭한 도구들은 무용지물이 되고 만다. 이런 면에서 IP는 문제를 표현하는 모형 언어적으로 심각한 제한을 지니고 있다.

본 논문에서는 CP의 제약식 확산법과 IP의 LP 기법이 새로운 모형 언어를 설계하는데 효과적으로 통합되어 새로운 문제 풀이의 솔버를 제공함을 논의하고자 한다. 즉 새로운 모형 언어의 틀에서 그 구조가 내포하는 성질을 통해 자연스럽게 두 기법을 통합하여 문제를 푸는 솔버 구축의 틀을 제시하고자 한다. 그 목적을 위해 우선 지금까지 이루어진 통합의 노력들을 거슬러 살펴보고, 조합 최적화 문제에서 IP와 CP의 특징들을 비교해본다. 그리고 셋째로 각 IP, CP의 문제 표현 모형을 예를 통해 비교해 보고 새로운 모형의 틀로 문제를 표현할 때 어떤 구조적인 특징들이 있는지를 살펴본다. 넷째로 새로운 모형의 구조를 이용해서 어떻게 새로운 솔버가 구현될 수 있는지를 살펴본다. 마지막으로 결론에서 논문을 요약 정리한다.

Ⅱ. 문헌 연구

제약식프로그래밍과 정수계획법(The Integer Programming, IP) 혹은 선형계획법(The Linear Programming, LP)을 통합한 하이브리드 솔버에 대한 아이디어는 최근 여러 논문에서 제시된 바 있다(Smith 등, 1995; Darby-Dowman, Little,1998). 그 논문에 따르면 각 사용하는 방법들이 갖고 있는 특징들에 따라 실험적인 결과들을 도출해 낸 바 있다. 특히 정수계획법은 좋은 이완문제(Relaxation Problem)을 갖는 경우에 효과적인 것으로 나타나 이완문제의 한계값이 약하거나 그 모형언어(Modeling Lanauage)에서의 약점으로 인해 크기가 큰 모형을 설정할 시에 문제 풀이에 많은 어려운 점들을 내포한다. 제약식 프로그래밍(Constraint Programming, CP)은 기존의 최적화 모형과는 달리 풍부한 표현 제약식들을 가지고 있어 모형

의 표현에 있어 크기가 상대적으로 작은 모형을 생산해낼 수 있고, 고도로 제약이 된 문제에서 강점을 드러낸다. 하지만 CP 이완법은 상대적으로 각 제약식의 국지적 특징들을 이용하였기 때문에 IP에 비해 전국적이지 못하다.

서로의 약점을 보완하기 위해 앞의 두가지 접근법을 통합하는 노력이 있어왔다.(Beringer, De Backer, 1995)는 CP의 한계 확산법(Bounds Propagation)과 고정 변수를 이용하여 CP와 LP를 결합하는 방법에 대해 생각해 본 적이 있다. (Rodosek 등, 1997)은 변수 도메인에 있는 값을 추려내고 값의 법위를 한정짓기 위해 한 탐색나무(Search Tree)에서 CP 솔버와 LP 솔버를 함께 사용하였다. 탐색나무의 각 노드에서 풀어야하는 하위문제는 몇 가지 경우에 실현 불가능성을 규정하는데, 확산법으로 한 변수의 도메인을 공집화하거나, LP 솔버를 통해 이완 문제가 실현 불가능함이 판명되거나, 이완문제의 목적함수 값이 현재 최적값보다 열등할 때가 그러하다. 그들의 아이디어는 문제를 CP로써 모형화하고 이 모형에 대해 체계적으로 "그림자" IP모형을 설정할 수 있는 방법을 프로시져로 만들었다. 그 방법은 재설정된 수식 제약식과 **alldifferent** 제약식 등을 포함한다. 그 위에 모들러(Modeler)가 구축되어 제약식들에 어떤 솔버를 - CP, LP 혹은 두가지를 동시에 - 이용할지를 결정한다. 상징형 제약식을 보다 잘 다루기 위한 결합을 목적으로 하는 연구도 진행되었다. (Hajian 등, 1996; Hajian, 1996)은 IP 솔버가 보다 잘 다룰 수 있는 방법을 보였다. 그리고 이를 통해 **alldifferent** 제약식을 선형모형에서 어떻게 다룰 지를 제시하였다. Bockmayr와 Kasper(Brockmyr, Kasper, 1998)은 CP와 IP를 통합하기위해 흥미로운 통합 틀을 선보였다. 그 틀에서 몇 가지의 통합이 가능한데, 어떻게 상징형 제약식이 절면 제약식처럼 IP에 통합이 가능한지, 선형 부등식 체계가 어떻게 상징형 제약식으로써 CP로 통합될 수 있는지를 보였고, 선형 부등식과 변수의 도메인이 같은 제약식 저장소(Constranit Store)에 저장될지를 논의했다.

Ⅲ. 조합 최적화(Combinational Optimization) 문제에서 CP와 IP의 특징

한정도메인 제약식프로그래밍(CP(FD) 혹은 FD)에서 각 정수 변수들은 초기에 최적값을 포함한 모든 가능한 값들을 수용하는 도메인 D_i를 가지고 있고, 이는 일반적으로 유한집합으로 표현된다. 그 도메인들의 직교곱(Cartesian Product) $D_1 \times ... \times D_n$은 주어진 문제의 해공간을 형성한다. 모든 조합문제의 해공간은 유한하기 때문에 이론적으로 모든 대안들을 탐색 평가함으로써 최적해는 얻어질 수 있지만, 실제적으로는 이 해공간은 탐색에만 의존할 수 없을 정도로 크므로 CP는 제약식으로부터 실현불가능해를 추론해 해공간을 축소해 간다. 즉 CP는 모든 가능한 대안으로부터 해의 가능성을 좁혀들어간다.

IP는 CP에서 변수의 도메인을 통해 정의되는 해공간을 축소해 나가는 대신, 탐색나무의 각 노드에서 정의되는 이완문제를 풀어 일련의 해들을 도출해 나간다. 이때 필요한 이완문제는 일반적으로 변수들이 정수 값을 갖게 하는 정수 제약식을 무시함으로써 선형계획 모형으로 얻어진다. 이외에도 주어진 문제의 구조나 제약식으로부터 얻을 수 있는 타당 제약식(Valid Constraints)을 모형에 추가함으로써 이완문제를 원 문제에 가깝게 조여줄 수 있다. 이렇게 얻어진 선형계획 모형은 선형계획법 알고리즘을 통해 푼다. 최적해를 찾기위해 이완문제를 통해 얻어지는 실현불가능 최적값과 실현가능 최적값은 원 문제 최적값이 한계값이 되어 탐색을 가속시킨다. 선형계획 이완문제의 쌍대문제는 정수 변수를 고정시키거나 추가시킨 제약식을 벤더스제약식(Benders Cuts)으로서 도출시킨다(Benders, 1962; Geoffrion, 1974). 이와 같은 정보가 본 논문이 제시하는 통합 틀에서 어떻게 이용될 수 있는지는 뒤에 더 상세히 논의된다.

CP는 다음과 같은 장치를 이용하여 해의 탐색을 가속시킬 수 있다.

- 실현 불가능성을 증명함을 통해 정수 변수의 도메인을 줄여간다.
- 한계 제약식(Bounds)과 정수계획법의 전통적인 제약식 등을 통해 원 문제에 더 가까운 이완문제를 갖게되고 이러한 제약식은 CP 솔버의 해 탐색을 가속할 수 있다.
- 대칭성을 통해서나 상징형 제약식이 갖고 있는 구조를 이용하여 탐색이 불필요한 가지를 잘라낼 수 있다.

반면 IP는 다음과 같은 방법을 통해 해의 탐색을 가속할 수 있다.

- 전국적 추론으로서의 선형계획 이완문제 풀이는 실현가능 해를 더욱 빨리 도출해 최적해의 탐색을 가속시킨다.
- 이완문제 해와 실현가능 해의 한계는 최적성의 증명을 가속시킨다.
- 실현 불가능 해로부터 얻어지는 정보는 실현 불가능성을 방지하는 방지 제약식(Nogoods)을 제공한다.

Ⅳ. CP, IP 그리고 하이브리드 솔버의 모형 비교

Ⅱ장에서 보았던 것처럼 지금까지 CP와 LP의 통합을 위해 택해진 방법들은 (가)CP와 IP의 모형을 함께 적용하여 양 솔버를 병렬적으로 이용하여 문제를 풀거나 (나) 한 방법을 다른 방법에 종속적으로 통합시키는 것이었다. 기존의 주어진 모형 틀을 사용하느냐 아니면 새로운 모형 틀을 사용하느냐는 아직 완전히 해결되지 않은 이슈이다. (가)의 접근 방식은 어떻게 하면 두 모형간의 결합을 강하게 유지하고 두 모형을 유지함으로서 나오게 되는 불필요한 중복요소를 해소하느냐가 관건이다. (나)는 한쪽을 포기함으로써 문제 해법에 제한적이다. 왜냐하면 CP의 추상적인 상징형 제약식은 IP의 모형에서 사용될 수 없고, IP의 절면(Cutting Planes)나 이완식(Relaxaion)들은 CP 모형에서 사용할 수 없기 때문이다.

CP, IP 그리고 하이브리드 모형을 비교하기 위해 다음의 간단한 다수기계의 스케줄링 예를 생각해보자. 이 예를 통해 새로운 혼합형 모형의 잇점이 드러날 것이다. 문제는 n개의 기계설비에서 수행해야 할 n개의 작업이 있을 때 이 작업들의 수행 스케줄을 어떻게 짜는 것이 좋은가하는 것이다. 각 기계 m은 r_m의 작업 속도를 보인다고 가정하자. 이때 해의 유불리를 측정하기 위한 목적치로서는 사용되는 기계의 고정 비용을 최소화하는 것으로 하자. 작업 j에 대해 R_j는 작업의 완성 시점이고 P_j는 속도 $r_m=1$에서 걸리는 작업 시간, D_j는 작업이 완성되어야하는 최후의 마감시간이다. C_m은 기계 m의 고정 비용이고 t_j는 작업의 시작 시간이다.

먼저 IP모형을 보자. 이 모형에서 작업이 수행되는 순서를 나타내기 위해 이진 변수 x_{ij}를 이용한다. 두 개의 서로 다른 작업 i와 j가 같은 기계에서 수행될 때 i가 j보다 시간적으로 앞서면 $x_{ij}=1$이고 그렇지 않으면 0의 값을 갖는다. 작업 j가 기계 m에 할당되면 이진 변수 $y_{mj}=1$ 그리고 기계 m이 사용되면 이진변수 $z_m=1$이다. 이때 모형은 다음과 같다.

$$\min \sum_m C_m z_m$$

$$s.t.$$

$$z_m \geq y_{mj} \qquad \forall m, j, \qquad (1)$$

$$\sum_m y_{mj} = 1, \qquad \forall j, \qquad (2)$$

$$t_j + \sum_m \frac{P_j}{r_m} y_{mj} \leq D_j, \qquad \forall j,$$

$$R_j \leq t_j, \ \forall j,$$

$$t_j + \sum_m \frac{P_j}{r_m} y_{mi} \leq t_j + M(1 - x_{ij}), \ \forall i \neq j, \qquad (3)$$

$$x_{ij} + x_{ji} \geq y_{mi} + y_{mj} - 1, \ \forall m, i < j \qquad (4)$$

$$z_m, y_{mj}, x_{ij} \in \{0,1\}, \ t_j \geq 0 \ \forall m, i, j.$$

제약식 (4)는 "big-M" 제약식이라 불린다. 이때 M

은 충분히 큰 수로 $x_{ij}=1$일 때 제약식은 작업 i가 작업 j를 선행할 것을 강제하고 $x_{ij}=0$이면 아무런 제약도 가하지 않는다.

다음은 같은 문제를 CP모형으로 설정한다. 여기서 변수 m_j는 작업 j에 할당된 기계를 의미한다.

$$\min \sum_{m} C_{m} z_{m}$$

s.t.

if $m_i = m_j$

then $(t_i + \frac{P_i}{r_{m_i}} \le t_j) \vee (t_j + \frac{P_j}{r_{m_j}} \le t_i), \quad \forall i, j,$ (5)

$$(t_j + \frac{P_j}{r_{m_j}} \le D_j), \quad \forall j,$$

$$R_J \le t_j, \quad \forall j,$$

if atleast $(m, [m_1, \ldots, m_n], 1)$

then $z_m = 1$ **else** $z_m = 0, \quad \forall m,$ (6)

$$m, m_j \in \{1, \ldots, n\}, \quad t_j \ge 0, \quad \forall j,$$

CP에서는 IP 모형에서 순열(Permutation)과 할당(Assignment)을 표현하는 두 가지 이진 변수, x_{ij}와 y_{mj}를 사용하지 않고 모형을 설정할 수 있다. 이 두 가지 변수들은 두 개의 첨자를 지니고 있어서 IP 모형의 크기를 확대하는 주된 요인이 된다. 따라서 이 변수들을 사용하지 않음으로 해서 CP는 모형의 크기를 줄인다. 이런 경우의 극단적인 예는 "진보적인 파티(Progressive Party Problem)"(Smith et al., 1995)에서 찾아볼 수 있는데 변수에 여러개의 첨자가 있음으로 해서 탐색해야 하는 해 공간을 극단적으로 크게 만든다.

IP 모형은 목적함수식 제약식 (1)-(2) 그리고 $0 \le y_{mj} \le 1$라는 선형 이완식을 갖는 잇점이 있다. y_{mj}는 모형의 크기를 확대하지만 이 이완식을 통해 그 약점을 상쇄한다. 하지만 순열 변수인 x_{ij}에 관련된 제약식은 약한 이완식을 만들면서 필요없이 모형의 크기를 확대한다. 따라서 혼합형 모형이 가져야 할 특징의 요점은 IP 모형과 같이 유용한 이완식을 도출할 수 있으면서 CP가 가진 모형의 간결성을 동시에 성취할 수 있느냐 하는 것이다. 이를 위해 한걸음 뒤로 물러서 기존의 전통적인 모형에 문제풀이 솔버들을 적용하기 보다는 솔버들에 적절한 모형을 어떻게 설계하는지에 관심을 둘 필요가 있다.

혼합형 모형의 틀로서 (Hooker, 1994)과 (Hooker, Osorio, 1999) 그리고 ((Hooker 등, 2001)에서 제기된 "혼합논리/선형계획법(Mixed Logic/Linear Programming, MLLP)" 방법을 살펴본다. 기본적으로 이 방법은 제약식을 문제의 이산형 부분과 연속형 부분을 추론을 나타내는 조건식의 형태로 표현하는 것이 그 핵심이다. 예를 들어 그 일반형을 다음과 같이 나타낼 수 있다

$$\min \ cx \qquad (7)$$

s.t.

$$h_i(y) \rightarrow A^i x \ge b^i, \quad i \in I,$$

$$x \in R^n, \quad y \in D,$$

여기서 y는 이산형 변수들의 벡터이고 x는 연속형 변수들의 벡터이다. 조건식의 전제(Antecedent)인 $h_i(y)$는 CP의 기법을 통해서 해결할 수 있는 부분이다. 조건식의 결과(Consequent)는 선형 부등식들로서 선형 이완식으로 흡수될 수 있는 부분이다. 전제없이 부가되어야 하는 선형 부등식 $Ax \ge b$이 있다면 이는 가상의 전제를 생각하여 $(0=0) \rightarrow Ax \ge b$와 같이 조건제약식으로 표현이 가능하다. 마찬가지로 선형 부등식 부분이 없이 부가되어야 하는 이산형 제약식이 있다면 이는 $\neg h \rightarrow (1=0)$과 같이 표현이 가능하다. 목적함수식에서 이산형 변수가 없는 것이 알고리즘 상에서 유익하다. 이산형 변수에 의존하는 목적함수 비용이 있을 때 이는 쉽게 조건 제약식으로 변환이 가능하다. 즉, 이산변수 x_j에 의존하는 목적

함수식 $\sum_j c_j x_j$가 있을 때 이는 새로운 연속변수 z_j를 도입하여 목적함수를 $\sum_j x_j$로 하고 $(x_j = 1) \rightarrow (z_j = c_j)$를 제약식으로 추가하는 것과 같다.

또한 유용한 모형 설정 도구로서 "변수첨자(Variable Subscript)"를 들 수 있다. 이것은 하나 이상의 이산형 변수를 포함하는 첨자를 의미하는 것으로 예를 들어 만약 c_{jk}가 노동자 k를 작업 j에 할당할 때 드는 비용이라 하면 할당에 대한 총 비용은 $\sum_j c_{jy_j}$로 y_j는 작업 j에 할당된 노동자를 의미하는 이산형 변수이다. c_{jy_j}는 사실상 $y=(y_1, \ldots, y_n)$의 함수로 $g_j(y)$와 같이 표현될 수 있다. 여기서 함수 g_j는 y_j에만 의존한다. 하이브리드 모형은 이 도구를 그 모형 틀에 다음처럼 통합 사용한다.

$$\begin{aligned} &\min \quad cx \\ &\text{s.t.} \\ &h_i(x) \rightarrow L_i(x, y), i \in I \\ &x \in R^n, y \in D \end{aligned}$$

여기서

$$L_i(x, y) = \sum_{k \in K_i(y)} a_{ik}(y)\ x_{j_{ik}(y)} \geq b_i(y)$$

이다. 모형에서 합의 기호에 "변수-인덱스-집합(Variable Index Set)" $K_i(y)$가 사용되었음에 주목하자. 이는 집합을 값으로 갖는 변수 y의 함수이다. 그리고 실수 값을 갖는 "변수상수(Variable Constant)" $b_i(y)$도 사용되었다. 이런 형태의 모형 역시 충분히 많은 조건 제약식을 추가함으로써 가장 기본적인 형태인 (7)로 표현 가능하다. 예를들어 제약식 $z \geq \sum_j c_{jy_j}$는 다음의 제약식을 추가하면 $z \geq \sum_j z_j$로 표현된다.

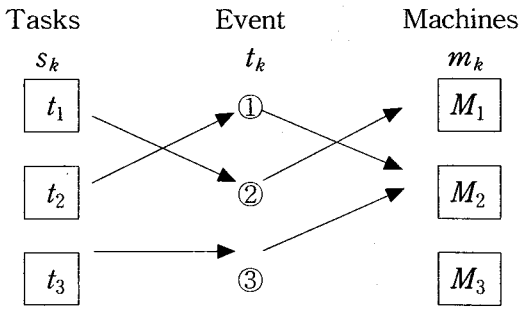
$$(y_i = k) \rightarrow (z_j = c_{jk}), \forall j, k \in \{1, \ldots, n\}$$

이때 각 $y_j \in \{1, \ldots, n\}$이다. 그러나 이런 변환보다는 가능하면 솔버 자체가 앞서 말한 도구들을 - 변수첨자나 변수-인덱스-집합 등 - 직접 다룰 수 있도록 하는 것이 바람직하다.

이제 앞에서 언급된 다수-기계-스케줄링 문제를 하이브리드 모형의 틀로 설정해 본다. 먼저 k를 이벤트(Event)들의 수열이라고 하자. 각 이벤트는 한 작업의 시작을 의미한다. 다음 모형에서 이벤트에 집중하여, 이벤트와 작업 그리고 이벤트와 기계들의 연결을 이용한다(그림1). 변수 t_k는 이벤트 k의 시작 시간이고, s_k는 시작할 작업, 그리고 m_k는 할당될 기계를 의미한다. 이때 하이브리드 모형은

$$\begin{aligned} &\min \quad \sum_m f_m \\ &\text{s.t.} \\ &(m_k = m_l) \rightarrow (t_k + \frac{P_{s_k}}{r_{m_k}} \leq t_l), \quad \forall k < l \\ &t_k + \frac{P_{s_k}}{r_{m_k}} \leq D_{s_k}, \qquad \forall k, \\ &R_{s_k} \leq t_k, \qquad \forall k, \\ &\textbf{alldifferent} \ \{s_1, \ldots, s_n\}, \\ &f_{m_k} = C_{m_k}, \ f_m \geq 0 \qquad \forall k, m. \end{aligned}$$

이 모형은 두 개의 첨자를 지닌 변수를 사용하지 않고 CP 모형의 간결함을 지니고 있다. IP에서와 같이 이완식을 얻기 위해서 단지 목적함수식과 (1)-(2)



〈그림 1〉 다수 기계 스케줄링에서 이벤트 작업 기계의 관계

을 모형에 포함시키고 변수 y_{mi}를 (10)에 있는 다른 변수들에 논리적으로 연결시킨다.

$$\min \quad \sum_m C_m z_m$$

s.t.

$$(m_k = m_l) \rightarrow (t_k + \frac{P_{s_k}}{r_{m_k}} \le t_l), \ \forall k < l$$

$$t_k + \frac{P_{s_k}}{r_{m_k}} \le D_{s_k}, \qquad \forall k,$$

$$R_{s_k} \le t_k, \qquad \forall k,$$

alldifferent $\{s_1, \ldots, s_n\}$,

$$z_m \ge y_{mj}, \qquad \forall m, j, \quad (8)$$

$$\sum_m y_{mj} = 1, \qquad \forall j, \quad (9$$

$$y_{m_k s_k} = 1, \qquad \forall k. \quad (10)$$

이제 이완식은 (8), (9), $0 \le y_{mj} \le 1$ 그리고 (10)가 모든 y_{mj}을 1에 고정시킴으로 y_{mj}라는 제약식하에서 $\sum_m C_m z_m$을 최소화시키는 것이다. 물론 y_{mj}와 t_k에 관련된 추가적인 제약식 이완식에 포함시킬 수 있다.

V. 하이브리드 솔버의 구현

5.1 알고리즘

앞 절에서 논의된 하이브리드 모형은 정수 계획법에서와 같이 이산 변수에 대한 가지치기(Branching)을 통해서 푼다. 모형의 조건 제약식은 CP와 LP의 역할을 분담시킨다. 즉 CP 솔버는 탐색나무의 크기를 줄이고 조건 제약식의 전제에 해당하는 이산형 제약식을 만족시키는 부분해(Partial Solution)을 찾기 위해 이산형 제약식에 적용이 된다. 탐색나무의 각 노드에서 LP 솔버는 $h_i(y)$가 "true" 값을 가지는 선형 제약식 $A^i x \ge b^i$를 만족하는 목적함수 cx를 최소화하다. 따라서 정수계획법에 비해 부등식의 적용이 연기되는 효과가 있다. 이런 부등식의 뒤늦은 적용은 LP 이완식을 작게 만들어 문제를 효율적으로 풀게 한다. 실현가능해는 각 전제가 "true"값을 갖게 되고 LP솔버가 주어진 부등식에 대해 최적해를 찾게 되면 얻어진다.

좀더 정확히 말하면, 알고리즘은 이산 변수 y_j값에 대한 가지치기에 의해 탐색을 함으로써 진행된다. y_j에 대해 가지치기가 이루어졌을 때 알고리즘은 각 가지를 y_j의 원 도메인 D_j를 서로 다른 부분집합 $\overline{D}_j$로 축소한다. 그리고 부분집합이 하나의 원소만을 지닌 집합으로 축소되었을 때 변수 y_j는 한 값에 고정된다.

탐색나무의 각 노드에서는 두 단계로 계산이 이뤄진다. 첫째로 제약식 프로그래밍의 솔버인 "체커"는 각 조건 제약식의 전제 $h_i(y)$에 대해 축소된 도메인이 그 전제를 만족하는지(Satisfying), 상충하는지(Violating), 혹은 결정하지 못하는지(Undetermined)를 본다. 이때 모든 $y \in \overline{D}_1 \times \ldots\ldots \times \overline{D}_n$가 $h_i(y)$를 충족시키면 그 도메인이 전제를 만족시키고 그렇지 못하면 상충된다고 정의한다. 예를들어 $\overline{D}_1 = \{1,2\}$이고 $\overline{D}_2 = \{1,2,3\}$라 하자. 이들 도메인은 $y_1 + y_2 \ge 2$은 만족시키고, $y_1 + y_2 \ge 6$과는 상충되며, $y_1 + y_2 \ge 4$에 대해서는 결정할 수 없다. 한 제약식은 만족되거나 상충되면 그 도메인에 의해 결정되었다(Determined)고 한다. 이를 판단하는 체커는 제약식이 결정되지 않으면 체커 역시 그렇다는 것을 보여야 한다는 측면에서 정확해야한다(Correctness). 그러나 때때로 제약식이 만족되는지 상충되는지를 보일 수 없다는 점에서 불완전(Incompleteness)하다. 하지만 모든 y_j가 고정될 때는 체커는완전해야한다(Completeness).

체커는 현재의 도메인이 제약식 $h_i(y)$를 만족하는지 상충하는지를 결정하기 위해 다양한 기법들을 사용한다. 제약식 만족(Constraint Satisfaction) 방법을 사용하는데 이 중 많은 부분이 제약식 확장법과 여타의 기법들에 의해 도메인을 축소하도록 설계

되었다. 즉, 그 방법들은, $h_{i_1}(y), \ldots, h_{i_n}(y)$을 $y \in \overline{D}_1 \times \ldots \times \overline{D}_n$에 의해 $L_i(x, y)$가 만족되지 않는 그런 조건 제약식의 전제들이라고 할 때에, 각각의 $i \in I$ 에 대해 $\{(y), \neg h_{i_1}(y), \ldots, \neg h_{i_k}(y)\}$에 적용이 된다. 그러므로 체커는 제약식 만족과 최적화 방법이 어떻게 통합될 수 있는가를 이해하는 일반적 방법을 제공한다.

두 번째 단계는 선형 제약식의 집합 L에 대해 목적함수식 cx을 최소화하는 선형계획 이완 문제를 풀기위해 LP 솔버(작은 의미에서 이 솔버)를 사용하는 단계이다. L은 모든 강제된 제약식 $L_i(x, y)$의 집합들의 합집합으로서 도메인 $\overline{D}_j$에 의해 만족되고 변수상수, 변수첨자와 변수인덱스 집합이 그 도메인에 의해 결정되는 그런 제약식들이다.

실현가능해는 각 $h_i(y)$와 강제된 제약식 $L_i(x, y)$가 결정적이고 솔버가 실현가능해를 찾았을 때를 말한다. 이 경우 y_j에는 각각의 도메인에서 어느 값이 할당된 상태이고 x는 LP솔버에 의한 최적해가 할당된 상태로 실현가능 해 (x, y)가 정의된다. 이런 y의 선택은 y가 목적함수식에 없기 때문에 그 영향을 받지 않는다.

실현가능해가 발견되거나 혹은 솔버에 의해 LP 이완문제가 실현불가능임이 밝혀지면 탐색은 백트랙(Backtrack)을 실시하고 그렇지 않으면 계속 가지치기를 이어 나간다. 자세한 알고리즘은 그림2에 나와있다.

(Hooker, Osorio, 1999)에서 보고된 계산 실험에서 MLLP 모형 틀은 모형 설정에서만 유리한 것이 아니라 전통적인 MILP 솔버보다 더 빠른 솔버를 제공한다. 그러나 그 연구에서는 몇가지 이슈들이 해결되지 않았는데

- 먼저 체계적인 변수첨자와 인덱스집합의 구현이 이루어지지 않았고,
- 탐색나무의 각 노드에서 LP 솔버를 통해 얻어진 해의 정보가 충분히 이용되지 않았고,
- 연속 변수에 대한 가지치기와 연속형 제약식에 대한 확산법(Propagtion)에 대한 구현과 이용이 없다는 점이다.

5.2 기본 구조

(Bockmyr, Kasper, 1998)에서 보여준 통합 틀은 여기에서 제시되는 통합 들을 보는 매우 흥미로운 시각을 제시한다. CP문헌에서는 원시(Primitive)제약식과 비원시(Non-primitive) 제약식을 구별하고 있는데, 원시제약식은 효율적인 실현가능이나 최적화 프로시져가 존재하는 (즉 다항식 알고리즘이 존재하는) "쉬운" 제약식을 의미한다. 이런 원시제약식은 제약식 저장고(Constraint Store)에서 관리 되는데, CP(FD)는 단순히 변수 도메인으로 구성된다. 비원시제약식의 확산 알고리즘은 제약식 저장고에서 현재의 도메인들을 추출하여 추론을 통해 더 작은 도메인을 만들어 내고 이를 다시 제약식 저장고에 저장한다. IP에서 연속형 변수에 대한 선형 부등식은 원시제약식이다. 왜냐하면 이들은 LP 솔버로 다루어질 수 있기 때문이다. 그러나 변수에 대한 정수 제약식은 비원시제약식이 된다.

(Bockmyr, Kasper, 1998)은 LP와 CP를 통합하는 두가지 방법을 제시하고 있다. 첫째는 주어진 문제의 LP 부분을 비원시 제약식으로 CP의 틀에 통합시킨다. 그래서 이 방법에서 LP 솔버는 제약식 확산법의 한 기법이 된다. 제약식 저장고에서 도메인은 유한구간의 형태로 접근되어 한 변수에 대한 최소화나 최대화를 통해 새로운 유계구간을 도출, 이를 다시 저장고에 저장한다.

둘째 방법은 선형 부등식을 원시 제약식으로 만드는 것이다. 제약식 저장고는 제약식들의 연속 부등식 이완식을 저장하고 정수 제약식은 배제한다. 예를 들어, 이산 제약식 $x_1 \vee x_2$와 $\neg x_1 \vee x_2$는 제약식 저장고에 $x_1 + x_2 \geq 1$과 $(1 - x_1) + x_2 \geq 1$ 그리고 $0 \leq x_j \leq 1$로 표현될 수 있다.만약 제약식 확산법이 $x_2 = true$를 연역해 내면 부등식 $x_2 \leq 1$이 저장고에 추가된다. 이 기법은 MILP에서 오랫동안 전처리기(Preprocessing)으로 알려진 기법들 중 한 예로 두 번째 통합 방법의 특수한 경우로 볼 수 있다.

1: $\overline{D}_j$: y_j의 현재 도메인

2: $L_i(x)$: $A^i x \geq b^i$

3: A: 활성 노드의 집합. 도메인 벡터 $\overline{D}=(\overline{D}_j)$로 정의된다. 최초값은 $\{D\}$이다.

4: $\overline{z}$: 최적치의 상한값. 최초로 ∞값을 갖는다.

5:

6: 모든 논리 제약식을 생성한다.

7: **while** $A \neq \emptyset$ **do**

8: 노드 $\overline{D}$를 A로보터 제거한다.

9: 제약식 확산법을 운용한다.

10: $L \Leftarrow \bigcup_{i \in I}\{L_i(x) \mid L_i(x)$ *is enforced by* $\overline{D}\}$

11: $z \Leftarrow \min\{cx \mid L\}$

12: **if** $z < \infty$ **then** {*L이 실현 가능하다.*}

13: **if** 모든 $\overline{D}_j$가 단일 원소 집합 **then**

14: **if** $z = -\infty$ **then**

15: stop(문제가 무계이다.)

16: **end if**

17: **if** $z < \overline{z}$ **then**

18: $\overline{z} \Leftarrow z$

19: $\overline{x} \Leftarrow \arg\min\{cx \mid L\}$

20: $\overline{y} \Leftarrow \{d_1, \ldots, d_n\}$ 여기서 $\overline{D}_j = \{d_j\}$이다.

21: **end if**

22: **else**

23: 노드 $\overline{D}^1, \ldots, \overline{D}^K$를 A에 추가함으로써 가지치기 한다. 이때 $\overline{D}_j^k \subset D_j$이다.

24: **end if**

25: **else** {*L은 실현불가능이다.*}

26: 방지 제약식을 생성한다.

27: **end if**

28: **end while**

29:

30: **if** $\overline{z} = \infty$ **then** {*실현불가능*}

31: 문제가 실현 불가능이다.

32: **else**

33: $(\overline{x}, \overline{y})$은 최적 유계해이다.

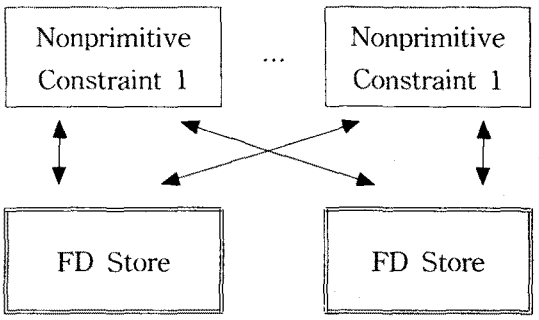
34: **end if**

〈그림 2〉 하이브리드 솔버의 알고리즘

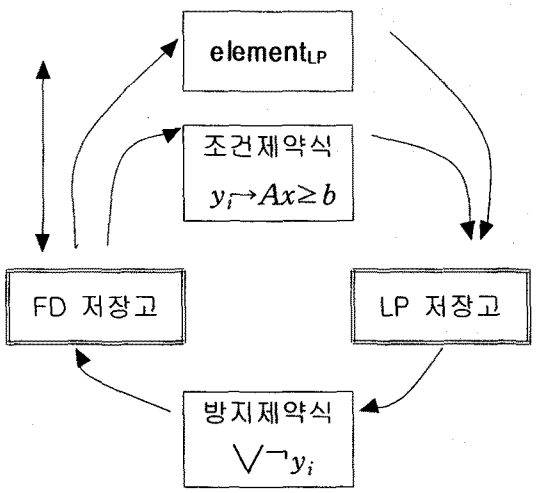
여기서 제시하는 통합 방법은 두 개의 제약식 저장고를 지니는 세 번째 형태로 볼 수 있다(그림3). 전통적인 FD 제약식 저장고 S_{FD}는 도메인을 포함하고 LP 제약식 저장고 S_{LP}는 선형 부등식과 유계구간을 포함한다. 비원시 제약식들은 양 제약식 저장고에 접근이 가능하며 또한 제약식을 추가 할 수도 있다. FD 저장고에서는 도메인 제약식 $x_i \in D_i$인 형태만 존재하므로 정수 제약식은 원시제약식으로 존재한다. 하지만 어떤 연속 변수도 FD 저장고에 존재하지 않으며 어떤 이산 변수도 LP 저장고에 존재하지 않는다. 하이브리드 모형의 조건 제약식은 두 저장고를 연결하는 핵심 추론자(Prime Inference Agents)로서의 역할을 한다. 즉 FD 저장고의 도메인을 읽고 LP 저장고에 부등식을 저장한다.(그림4).

(Hooker, Osorio, 1999)의 MLLP에서 조건제약식은 한쪽 방향으로만 적용이 된다. 즉 S_{FD}의 데이터로부터 추론하여 S_{LP}에 새로운 데이터를 추가하지만 그 반대 방향은 성립하지 않는다. 그 이유는 알고리즘이 이산변수에 대해 가지치기르 하고 있기 때문이다. 가지치기를 통해서 이산 변수의 도메인이 줄어들면 제약식 확산법에 의해 더 많은 조건 제약식이 전제들에 대한 "true"값이 추론되고 더 많은 선형 부등 제약식이 LP 저장고에 추가된다. 반면 조건 제약식의 반대 방향의 추론은 연속구간을 도메인으로 가지고 있는 연속 변수에 대해 그 연속구간을 분할하는 가지치기를 함으로써 가능하다. 조건제약식의 전제는 연속수치 제약식이고(반드시 선형 부등식일 필요는 없다) 그 결과는 원시 이산 제약식 즉 이산 변수의 도메인에 대한 제약식이다. 이 경우 전제의 진위값은 연속구간 확산법(Interval Propagation)에 의해 추론된다.

다음으로 하이브리드 모형의 비원시제약식의 두가지 예를 검토해 본다. 그 첫째는 변수첨자를 구현하는데 사용되는 일반적인 형태의 **element** 제약식, 그리고 둘째로 S_{LP}에서 방지 제약식(Nogoods)을 도출하는 제약식이다. 이를 통해 하이브리드 솔버를 구현에 필요한 두가지 메카니즘을 보인다.



〈그림 3〉 하이브리드 솔버의 제약식 저장고와 비원시제약식



〈그림 4〉 하이브리드 솔버의 비원시 제약식들

5.3 변수 첨자의 구현

앞에서 본 것처럼 하이브리드 모형은 모형 구성요소로서 변수 첨자를 제공하지만 조건제약식으로의 확장은 대부분의 경우 힘들다. 대신에 그런 변수첨자를 다루기 위해 어떤 비원시제약식 혹은 추론자를 설계함을 통해 가능하다.

기본적으로 변수첨자가 생기는 두가지 경우가 있다. 하나는 이산 제약식에서이고 다른 하나는 연속 부등식에서다. 전자의 경우 변수첨자는 상수의 벡터이거나 이산 변수의 벡터일 수 있다. 그리고 이 두 경우 모든 주요 CP체계나 라이브러리(예로(Dincbas 등., 1988; SICS, 1995))에서 발견할

수 있는 **element/3**((Marriott, Stuchey, 1998))를 사용하는 것과 동일하다. 이 제약식은 **element**$_{FD}$ $(I, [X_1, \ldots, X_n], Y)$의 형태를 취한다. 이때 I는 도메인 $D_I = \{1, \ldots, n\}$을 가진 정수 변수로 다음에 나타나는 리스트의 인덱스이고 $[X_1, \ldots, X_n]$은 Y가 취할 값들의 리스트이다. 따라서 $Y = X_I$가 성립한다.

다음으로 두 번째 경우 **element**$_{LP}$ $(I, [X_1, \ldots, X_n], Y)$를 생각해 보자. 여기서 I는 여전히 이산형 인덱스 변수이다. 그러나 X_i와 Y는 연속변수나 상수이다. 이 제약식에 대한 확산은 거의 전과 동일하다. 즉 연속구간 $[\min(x_i), \max(x_i)]$를 x_i의 도메인이라고 하자. 그때 I의 도메인의 변화에 대해

$$\min = \{\min(x_i) \mid i \in D_I\}$$

$$\max = \{\max(x_i) \mid i \in D_I\}$$

의 값을 S_{CP}에서 D_I를 읽음으로써 계산할 수 있다. 그 두 값에서 $\min \le Y \le \max$

연속구간을 통해 S_{CP}에 첨가할 수 있다. 비슷한 방식으로 Y의 상한값과 하한값은 도메인 D_I를 줄이는데 사용된다. (LP에서 더 강한 상한 하한값을 S_{LP}에 있는 선형 부등식에 대해 그 변수를 최소화나 최대화시킴으로써 얻어질 수 있다. 이 방법은 어떤 경우에 유용할 수 있다.)

여기서 중요한 점은 이 제약식을 어떻게 확산하는가를 자세히 기술하는 것보다는 어떻게 추론자가 두 제약식을 저장고에 접근하는지를 예로써 보임으로써 앞의 하이브리드 솔버의 구조가 타당성을 갖는지를 보여주는 것이다.

5.4 실현 불가능 LP를 이용한 확산

LP 이완식을 풀은 후 이것이 실현 불가능으로 판명되었을 때 쌍대 해는 LP의 제약식의 실현 불가능 선형결합을 나타낸다. 조건 제약식 $y_i \rightarrow A^i x \ge b^i$, $i \in I$를 생각해보자. 여기서 어떤 $ax \ge b \in A^i x \ge b^i$는 0이 아닌 쌍대해를 가지고 있다. 이때 논리 제약식

$$\bigvee_{i \in I} \neg y_i$$

을 설정할 수 있다. 이것은 일종의 방지 제약식(Nogood)(Tsang, 1993)인데, 해당 선형 부등식 집합이 실현불가능 결합을 나타내므로 주어진 문제의 모든 해는 위의 논리 제약식이 부과하는 조건을 만족시켜야 한다. 이렇게 실현 불가능 LP로부터 추가적인 방지제약식을 추론하는 기법은 S_{LP}에서 데이터를 읽어서 S_{FD}에 기록하는 nogood이라는 이름의 비원시제약식으로 캡슐화할 수 있다. 그리고 이런 추론자는 탐색나무의 실현불가능 LP이완식을 갖는 노드에서 0이 아닌 쌍대값을 어떤 조합에 대한 방지제약식으로 모으고 통합하고 유지하는 역할을 수행하며 동시에 원시, 비원시제약식을 추론하여 S_{FD}에 저장된 데이터를 강화시키는 일을 수행한다. 실현불가능 조합의 사용에 관련하여서는 "똑똑한 백트랙킹(Intelligent Backtracking)"이라는 문맥 하에서 연구된 바 있다.(De Backer, Beringer, 1991). 그림4는 하이브리드 솔버에서 방지 제약식을 비롯한 기본적인 비원시제약식을 보여준다.

5.5 실현가능 LP를 이용한 확산

IP에서 탐색나무 각 노드의 이완식의 해는 모든 변수들에게 값을 정해준다. 그러나 하이브리드 솔버에서는 이산 변수의 값은 정해지지 않고 단지 도메인으로 나타난다. 왜냐하면 하이브리드 솔버에서는 연속변수에 관련된 제약식만을 가지고 이완식을 풀기 때문이다. 하지만 이런 이완식이 실현가능일 때 연속 변수들의 값인 이완식의 해 $\bar{x}$에서 이산 변수의 값으로까지 휴리스틱을 통해 확장이 가능하다. 그리고 이렇게 얻어진 해 $(\bar{x}, \bar{y})$는 모든 제약식을 만족시킨다. 이때 이산 변수 y는 목적함수식에 나타나지 않으므로 y의 확장된 값은 최적치에 영향을 주지 않는다.

조건 제약식 $h_i(y) \rightarrow A^i x \geq b^i$이 주어졌을 때 만약 $A^i \bar{x} \leq b^i$이면 $h_i(\bar{y})$는 "false"가 되어야 하고 $A^i \bar{x} \geq b^i$이면 "false"일수도 "true"일 수도 있다. 따라서 y_i의 현재의 도메인에서 $\bar{y_i}$의 값을 할당하는, 그래서 "false"여야 하는 조건 제약식의 전제를 "false"로 만들어 줄 수 있는 휴리스틱을 사용할 수 있다.

Ⅵ. 결 론

IP와 CP는 오랜동안 서로 다른 영역에서 서로 다르게 사용되어 왔지만 최근에서 서로의 장점들을 인식하기 시작했고 미래의 최적화 문제의 틀에서 서로를 보완적인 기법으로 인식, 그 통합의 방법을 모색하고 있다. 하지만 그 통합은 단순한 솔버 기법의 결합이 아닌 각자 수리계획법과 제약식프로그래밍 영역내의 모형화 전통 역시 새로운 관점에서 통합되어야 한다. 이 논문은 새로운 모형의 틀의 관점을 소개하고 그 틀에서 해법의 틀이 어떻게 구성되는지를 살펴보았다. 연속형과 이산형의 제약식은 조건제약식 아래에 자연스럽게 통합되었고 서로 다른 분야에서 사용되는 두 솔버를 깨끗하게 분리하면서 CP 솔버가 다루는 제약식과 LP 솔버가 다룰 수 있는 선형 부등식을 자연스럽게 연결시켜 문제 해결 능력을 상승시킬 수 있는 방법을 제시하였다.

참 고 문 헌

Benders, J. F., "Partitioning procedures for solving mixed-variables programming problems", *Numer. Math.*, 4: 238-252, 1962.

Beringer, H. and B. De Backer, "Combinatorial problem solving in constraint logic programming with cooperating solvers", *Logic Programming: Formal Methods and Practical Applications* (C. Beierle and L. Plumer eds.), Studies in Computer Science and Artificial Intelligence, Elsevier 1995.

Bockmayr, A. and T. Kasper, "Branch-and-infer: A unifying framework for integer and finite domain constraint programming", *INFORMS J. Computing*, 10(3): 287-300, 1998.

Darby-Dowman, K. and J. Little, "Properties of some combinatorial optimization problems and their effect on the performance of integer programming and constraint logic programming", *INFORMS J. Computing*, 10(3): 276-286, 1998.

De Backer, B. and H. Beringer, "Intelligent backtracking for CLP languages: an application to CLP(R)", *Ligic Programming, Proceedings of the 1991 International Symposium* (V. Saraswat and K. Ueda eds.), 405-419, San Diego, The MIT Press 1991.

Dincbas, M., P. Van Hentenryck, H. Simonis, A. Aggoun and F. Berthier, "The Constraint Logic Programming Language CHIP", *Proceedings International Conference on Fifth Generation Computer Systems(FGCS- 88)*, 693-702, Tokyo, ICOT, 1988.

Geoffrion, A., "Lagrangian relaxation for integer programming", *Mathematical Programming Study* 2: 82-114, 1974.

Hajian, M., R. Rodosek and B. Richards, "Introduction of a new class of variables to discrete and integer programming problems", *Baltzer Journals*, 1996.

Hajian, M., "Dis-equality constraints in linear/integer programming", Technical Report, IC-Parc, 1996.

Hooker, N. and M. Osorio, "Mixed logical/linear programming", *Discrete Applied Mathematics* 96-97 (1-3):395-442, 1999.

Hooker, J., H. Kim, G. Ottosson, "A declarative modeling framework that integrates solution methods", *Annals of Operations Research*, 104(1): 141-161,

Baltzer Science, 2001.

Hooker, J., "Logic-based methods for optimization", *Principles and Practice of Constraint Programming, PPCP'94* (A. Borning ed.), Lecture Notes in Computer Science Vol. 874, Springer, 1994.

Marriott, K. and P. Stuckey, *Programming with Constraints: An Introduction*, MIT Press, 1998.

Rodosek, R., M. Wallace and M. Hajian, "A new approach to integrating mixed integer programming and constraint logic programming", *Baltzer Journals*, 1997.

Smith, B., S. Brailsford, P. Hubbard and H. P. Williams, "The prograssive Party Problem: Integer Linear Progamming and Constraint Programming Compared", *Proceedings 1st International Conference on Principles and Practice of Constraint Programming, CP95*, 1995.

Tsang, E., *Foundations of Constraint Satisfaction*, Academic Press, 1993.

Swedish Institute of Computer Science, *SICStus Prolog User's Manual*, URL http://www.sics.se/sicstus, 1995.

Information System Review
Volume 5 Number 2
December 2003

On Implementing a Hybrid Solver from Constraint Programming and Optimization

Hak-Jin Kim*

Abstract

Constraint Programming and Optimization have developed in different fields to solve common problems in real world. In particular, constraint propagation and linear programming are their own fundamental and complementary techniques with the potential for integration to benefit each other. This intersection has evoked the efforts to combine both for a solution method to combinatorial optimization problems. Attempts to combine them have mainly focused on incorporating either technique into the framework of the other with traditional models left intact. This paper argues that integrating both techniques into an old modeling frame loses advantages from another and the integration should be molded in a new framework to be able to exploit advantages from both. The paper propose a declarative modeling framework in which the structure of the constraints indicates how constraint programming and optimization solvers can interact to solve problems.

Keywords : Constraint Programming, Constraint Satisfaction, Combinatorial Optimization, Constraint Propagation, Linear Programming, Integer Programming, Solver Technology

* Assistant Professor, School of Business, Yonsei University

◐ 저 자 소 개 ◑

김 학 진(hakjin@yonsei.ac.kr)

저자는 2001년 12월 미국 펜실베니아주 피츠버그에 있는 카아네기멜론 대학 GSIA 경영대학원을 OR 전공으로 졸업하여 박사 취득, 현재 연세대학교 경영대학 경영학과 조교수로 재직 중입니다. 저자의 관심분야는 최적화, 제약식프로그래밍, 그리고 이 둘을 결합한 통합 솔버의 구현에 관심이 있으며, 이들 기법을 이용한 SCM 스케줄링, 제품 설계, 최적 포트폴리오 선택, 통신 경영 등에서 나올 수 있는 제반 응용문제에 관심을 두고 있습니다.