효율적인 유지보수 작업배정을 위한 CSP기반의 스케줄링 시스템

심명수* · 조근식**

Scheduling System for Effective Assignment of Repair Warrant Job in Constraint Satisfaction Problems

Moung-Soo Shim* · Geun-Sik Jo**

Abstract

오늘날의 기업은 상품을 판매하는 것 뿐만 아니라 기업의 신용과 이미지를 위해 그 상품에 대한 사후처리(After Service) 업무에 많은 투자를 하고 있다. 이러한 양질의 사후서비스를 고객에게 공급하기 위해서는 많은 인력을 합리적으로 관리해야 하며 요청되는 고장수리 서비스 업무의 신속한 해결을 위해 업무를 인력에게 합리적으로 배정을 해야 한다. 그러므로, 회사의 비용을 최소화하면서 정해진 시간에 요청된 작업을 처리하기 위해서는 인력들에게 작업을 배정하고 스케줄링하는 문제가 발생된다.

본 논문에서는 이러한 문제를 해결하기 위해서 화학계기의 A/S작업을 인력에게 합리적으로 배정하는 스케줄링 시스템에 관한 연구이다. 먼저 스케줄링 모델을 HP사의 화학분석 및 시스템을 판매, 유지보수 해주는 Y사의 작업 스케줄을 분석하여 필요한 도메인과 고객서비스전략과 인력관리전략에서 제약조건을 추출하였고 여기에 스케줄링 문제를 해결하기 위한 방법으로 제약만족문제(CSP) 해결기법인 도메인 여과기법을 적용하였다. 도메인 여과기법은 제약조건에 의해 변수가 갖는 도메인의 불필요한 부분을 여과하는 것으로 제약조건과 관련되어 있는 변수의 도메인 크기가 축소되는 것이다. 또한, 스케줄링을 하는데에 있어서 비용적인 측면에서의 스케줄링방법과 고객만족도에서의 스케줄링 방법을 비교하여 가장 이상적인 해를 찾는데 트래이드오프(Trade-off)를 이용하여 최적의 해를 구했으며 실험을 통해 인력에게 더욱 효율적으로 작업들을 배정 할 수 있었고 또한, 정해진 시간에 많은 작업을 처리 할 수 있었으며 작업을 처리하는데 있어 소요되는 비용을 감소하는 결과를 얻을 수 있었다.

Keywords: CSP(Constraint Satisfaction Problems), Dynamic Scheduling, Customer Satisfaction Index

* 인하대 전자계산공학과
** 인하대 전자계산공학과 부교수

1. 서 론

A/S작업배정 스케줄링 문제는 한정된 인적자원을 가지고 동적으로 변화하는 분산된 장소에 발생된 작업을 배정하는 문제이다. A/S작업배정 스케줄링을 하는 전문가가 사용하는 여러 제약 조건들을 추출하여 스케줄링에 반영되어야 하며 이러한 작업배정문제는 일반적인 스케줄링문제로 n개의 업무를 m개의 인력에게 배정하는 O(n!×m)의 복잡도를 갖는 NP 문제이다. 이 스케줄링 문제는 각 인력의 근무가능시간대를 고려하여 서비스 사용자와 서비스 공급자가 모두 만족하는 윈윈 전략(Win-Win Strategy)의 해를 찾아야 하며 서비스를 공급하는 업무의 특성을 고려할 때 서비스 사용자의 시간적 욕구를 만족하며 서비스 공급자의 인력 관리비용을 모두 고려하는 추론 메카니즘의 기능도 제공되어야 한다. 또한, 한정된 인력을 가지고 사용자에게 양질의 서비스를 제공하기 위해서는 서비스사용자의 시간적인 만족을 충족시켜주는 재스케줄링 기능 또한 가지고 있어야 하며 서비스 공급자의 정책에 따라 비용에 대해서도 만족을 시켜주는 기능을 가지고 있어야 한다. 이러한 어려움으로 인해 현업에서는 A/S작업배정문제를 스케줄링 하기에는 전문가의 순간적인 판단에 전적으로 의존되므로 고비용과 비합리성이 도출되는 스케줄이 되고 있다.

전통적으로 연구자들은 이러한 스케줄링 문제를 풀기 위해 수리계획법을 사용해 왔으나 이와 같은 문제를 수학적 변수로 이루어진 도메인과 제약조건으로 모델링하기에는 무리가 있으며 해를 구하는데도 오랜 시간이 소비된다[Dhar90] [Schrijver 93]. 그러나, 근래의 많은 연구자들은 이러한 문제를 CSP (Constraint Satisfaction Problems : 제약조건 만족 문제) 해결기법과 같은 인공지능 방법을 제안하고 있다[Tsang 93]. 복잡한 스케줄링 문제를 모델링 하는 데에 있어서는 수리계획법보다 제약만족기법이 더 나은 융통성과 효율성을 제공하고 있기 때문이다. 이러한 인공지능 기법은 여러 분야에 접목되어지고 있으며 스케줄링 문제에 인공지능 기법을 적용시킨 많은 전문가 시스템이 소개되고 있다. RACES(Ramp Activity Coordination Expert System)라는 시스템은 많은 항공기들을 복잡한 제약조건하에서 주기장에 주기시키는 스케줄링 시스템으로 현업에 적용되어 사용되어지고 있다[Jo97].

본 논문에서는 먼저 스케줄링 모델을 현업에서 사용되어진 업무 스케줄을 분석하여 필요한 도메인과 고객서비스전략과 인력관리전략에서 제약조건을 추출하였다. 여기에 스케줄링 문제를 해결하기 위한 방법으로 CSP의 대표적인 해결기법인 백트래킹(Backtracking)과 제약조건 네트워크(Constraint Networks)의 일치성 검사를 적용하여 문제를 모델링 하였으며 여러 가지 제약조건을 만족하는 해를 구하는 A/S작업배정 스케줄링 전문가 시스템을 구현하고자 한다.

2. 이론적 고찰

2.1 CSP의 구성요소 및 해결방법

A/S작업배정 스케줄링 문제는 그 문제의 특성상 제약만족문제로 적용하기가 쉽고 이러한 접근은 기존의 규칙기반의 접근보다 좀더 속도적인 측면에서 효과적이다. CSP는 여러 가지 제약조건을 만족하는 해를 구하는 것으로 CSP의 구성요소는 다음과 같다.

1) 변수들의 집합 : $V=\{v_1, v_2, v_3, ..., v_n\}$

2) 각 변수Vi가 가질 수 있는 도메인 : D = {d_1, d_2, d_3,..... d_n}
3) 이러한 변수들 또는 변수간의 제약조건 : C= {c_1, c_2, c_3,..... c_n} 으로 이루어진다.(단, n은 변수의 개수)

이러한 문제에서 각 변수를 전향검사(Forward Checking)를 이용하여 제약조건의 일치성을 검사하고 이에 따라 도메인 여과를 적용하여 이것에 근간을 두어 백트레킹으로 할 경우에는 전체 탐색공간의 크기를 줄 일수 있다[Mackworth97] [Frost94].

CSP 해결기법에서 가장 효과적인 방법 중에 하나는 도메인 여과기법(Domain Filtering)이다. 도메인 여과기법은 제약 조건에 의해 변수가 갖는 도메인의 불필요한 부분의 여과, 즉, 그 제약조건과 관련되어 있는 변수의 도메인의 축소가 된다. CSP에서 도메인의 여과는 제약조건이 충돌되는 부분에 있는 도메인을 제거하는 것이다. 이러한 도메인의 축소는 대단한 효과를 가져온다. 기존의 규칙기반으로 문제를 해결할 경우에는 도메인의 여과에 의해 탐색공간을 축소하지 못했기 때문에 문제를 해결하는 데 필요 이상의 시간이 요구되어졌다. 그러나 도메인의 여과기법을 이용하는 제약 만족문제에 있어서는 도메인의 축소, 즉, 탐색할 공간이 축소되어 문제 해결을 위한 시간을 효과적으로 줄일 수 있게 된다. 따라서 해를 구할 경우에 불필요한 백트래킹(Backtracking)을 하지 않게 되므로 보다 효과적이다[Wallace94].

2.2 일치성 검사

불필요한 도메인을 제거하는 도메인 여과기법은 제약조건의 일치성 검사(Constraint Consistency Check)로서 이루어진다. 일치성 검사는 제약조건의 불일치성을 일으키는 부분을 찾아 불일치를 유발하는 도메인을 제거함으로써 일치성을 유지하는데 이러한 일치성 검사를 위하여 일반적으로 제약조건 네트워크를 구성한다. 제약조건 네트워크는 제약조건이 적용된 노드(Node)에서 아크(Arc)를 거쳐서 제약조건을 다른 노드에 전파하여 충돌된 제약조건의 도메인을 제거함으로써 제약조건의 일치성을 유지한다. 제약조건의 전파는 제약조건이 적용된 변수의 도메인에서 제약조건의 변경이나 도메인의 바인딩(Binding)으로 인하여 그 변수의 도메인 변형이 성공적으로 이루어지면 제약조건의 일치성을 유지 시켜준다고 할 수 있다. 이러한, 제약조건 전파로 인한 네트워크 상에서의 제약조건의 일치성은 다음3가지로 구분되어 검사한다 [Wallace94].

1) 노드에 대한 일치성(Node Consistency)

모든 변수에 대해서 $x \in D_i$,$P_i(x)$를 만족하면, 노드 i는 노드 일치성을 이룬다. (단, $P_i(x)$는 단항 일치)

2) 아크에 대한 일치성(Arc Consistency)

arc(i, j)는 $x \in D_i$ 인 x 에 대해서 $P_i(x)$일 때, $y \in D_i$인 어떤 y에 대해서 $P_j(y)$를 만족할 때, $P_{ij}(x,y)$ 를 만족하면 아크 일치성을 이룬다.

3) 패스에 대한 일치성(Path Consistency)

길이가 m인 노드 (i_0, i_1, i_2,..., i_m)에서 $x \in D_{i0}$이고 $y \in D_{im}$인 모든 x, y에 대하여 $P_{i0}(x)$, $P_{im}(y)$, $P_{i0\ im}(x,y)$일 때, $z \in D_{i1}$,...,$z \in D_{im-1}$에 대해서

(i) $P_{i1}(z_1)$and ...and $P_{im-1}(z_{m-1})$
(ii) $P_{i0\ i1}(x,z_1)$and $P_{i1\ 2}(z_1,z_2)$and...and$P_{im-1\ im}(z_{im-1},y)$

를 만족하면 패스에 대한 일치성을 이룰 수 있다. 그러므로, CSP를 이용하여 문제를 해결하기 위해서는 그 문제에 맞게 변수들과 변수 내에 존재하는 값들, 그리고 각 변수들에 대해서 값들이 만족시켜야 하는 제약조건을 정하여 네트워크를 구성한 후 구성된 제약조건 네트워크에 대해서 일치성 검사 알고리즘을 적용하여 각 변수에 남아있는 조건에 만족하지 않는 값들을 제거해야 한다. 이러한 값들을 모두 제거하게 되면, 네트워크는 일치성을 이루게 되고 일치성을 이룬 네트워크에 대해서 백트래킹 기법을 이용하여 해를 찾아가게 되는 것이다. 또한 CSP에서의 제약조건에는 2가지 경우로 나누어 생각할 수가 있는데 단항 제약조건(Unary Constraints)은 오직 하나의 변수에 영향을 미친다. 이 제약조건은 초기에 변수의 도메인으로 부터 제약조건을 만족하지 않는 값들을 제거하는데 사용될 수 있다. 예를 들면 제약조건 X≠1이 있다고 하면 변수 X의 도메인에서 1은 제거될 수 있다. 그러면 제약조건은 만족 되는 것이다. 이러한 단항 제약조건은 영향을 받는 변수의 도메인을 프리프로세싱을 통해 처리 할수 있으므로 프리프로세싱 후에는 무시될 수 있는 제약조건이다. 이항 제약조건(Binary Constraints)은 두개의 변수에 영향을 미친다. CSP의 모든 제약 조건들이 이항 제약조건이면 변수들과 제약 조건들을 제약조건 그래프(Constraint Graph)로서 나타낼 수 있다. 그래프의 각 노드는 변수를 나타내고 노드사이의 간선은 두 변수 사이의 제약조건을 나타낸다. 사실 이항 이상의 제약조건들은 모두 이항 제약조건으로 표현될 수 있기에 모든 CSP는 이항 CSP로 나타낼 수 있는 것이다. 도메인의 크기가 m1과 m2인 두 변수 사이의 이항 제약조건은 0 또는1의 값을 가지는 m1×m2의 행렬로 나타낼 수 있다. 1은 두 변수간의 제약조건을 만족하는 의미이며 0은 그렇지 않음을 나타낸다.

2.3 CSP문제의 정형화

CSP기반의 접근 방법을 어떠한 문제에 적용하기 위해서는 해결하고자 하는 문제를 CSP로 정형화하는 것이 필수적이다. 모델링 방법은 여러 가지가 있을 수 있으며 각 방법에 따라 탐색 공간의 크기가 달라지기 때문에 문제 해결에 상당한 성능의 차이를 발생시킬 수도 있다. 그러므로 효율적인 모델링을 위해서는 상당한 노력이 요구되어진다. PERT문제에서의 CSP정형화를 예를 들면 PERT문제는 우선순위 제약조건(Presidency Constraints)을 가지는 문제로서 N개의 태스크로 구성되며 각 태스크는 완료되기 위해 일정한 소요기간이 필요하며 다른 태스크에 대해 우선순위가 존재한다. 문제를 정형화하면 도메인 변수들이 가지는 도메인의 범위는 N개의 태스크를 모두 수행하는 경우의 최대 작업기간인 태스크의 소요기간(D)의 합(P)이다.

$$\sum_{i=1}^{n} D(i) = P$$

도메인의 범위가 결정되고 나면 각 도메인 변수(V)의 도메인이 결정된다.

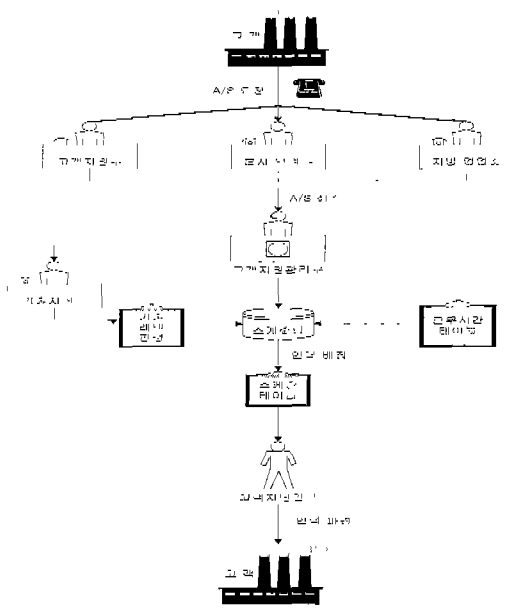
$$1 < \sum_{i=1}^{n} V(i) < P$$

다음으로 우선 순위 제약조건을 지정해 주면 문제의 정형화가 이루어진다. 이와 같은 CSP모델링은 제약만족기법을 적용하여 시스템을 구축하는데 있어서 가장 핵심이 되는 것이다. CSP기법으로 스케줄링문제의 해를 구하는 이유는 정의된 변수들은 주어진 문제에 있어서 개체에 직접 대응되고 제약조건을 수학적 표현으로 정의하지 않고도 표현 될 수 있으며 간단한 CSP 알고리즘이더라도 정수 프로그래밍 기법보다 해를 빨리 구할 수 있기 때문이다[Nadel89].

3. A/S작업배정 스케줄링 시스템

3.1 A/S 작업배정 업무에 대한 분석

A/S작업배정 스케줄링 시스템은 서로 다른 단계의 기술을 보유한 한정된 인력에게 분산된 현장에서 요청되는 A/S 작업을 배정하는 것으로



〔그림 1〕 A/S작업배정 업무의 시스템 흐름도

여러 제약조건을 만족시키면서 합리적으로 작업을 배정하는 시스템이다. 인력은 각각의 정해진 근무시간 동안 제약조건을 만족시키는 작업을 배정받아야 되며, 각 인력들의 기술습득 수준도 다양하여 기술습득 수준에 따라 기술인력의 비용을 10단계로 나누고 있다. 배정된 작업의 현장도 여러 개의 지역으로 나누어져 있으며 각각의 현장은 다양한 단계의 기술수준을 가진 인력을 요구한다.

이러한 여러 가지 제약조건을 고려하여 합리적인 최적의 스케줄링을 하는 것이다. [그림1]은 이러한 A/S작업배정업무의 전체적인 흐름을 보여주고 있다.

3.2 A/S 작업배정 업무현황

1. 인력은 S/W, H/W 부분으로 인력이 나누어지며 각 부분은 기술교육 및 경력, 지식수준에 따라 10단계로 구분된다.
2. 인력은 각자의 근무시간표를 주별 또는 월별로 작성하여 제출한다.
3. 처리한 작업의 양에 비례하여 기본급에 수당을 지급한다.
4. 인력은 하루에 평균 4-7건의 일을 처리한다.
5. 평균 2일간의 작업배정 스케줄표가 작성되지만 1일을 기준으로 작성 되어진다.
6. 초급단계의 기술을 요구하는 작업에서 고급단계의 기술을 요구하는 작업으로 갈수록 요구되는 작업의 빈도수가 줄어든다.
7. 특정고객의 경우 방문시간을 지켜야만 업무처리가 가능하다.
8. 조직 내에는 사후처리업무에 약 120명의 직원이 있고 이는 80명이 하드웨어 부분, 40명이 소프트웨어 부분으로 나누어진다.
9. 영업소는 서울, 대전, 군산, 대구, 부산, 포항, 울산, 광양에 8개소가 있으며 영업소마

다 담당하고 있는 업무처리지역으로 나누어진다.

10. 하루동안 처리되는 업무건수는 300 - 440건이다.
11. 업무처리 시간은 30분에서 4시간이 소요되며 평균 소요시간은 2시간이다.
12. 각 고객지원팀의 5명 전문요원이 1또는 2일전에 스케줄을 잡는다.

3.3 A/S작업배정 스케줄링 시스템

3.3.1 도메인의 분석 및 정의

작업에 대한 도메인들의 분석은 사후처리 업무를 고객으로부터 접수 할 때 작성되는 작업지시서에 기록되는 사항을 분석하여 도메인으로 정의하였고 인력에 대한 도메인 분석은 각 인력의 기술인사 사항과 매주 제출되는 근무시간표를 기초로 하여 작성되었다.

1) 작업에 대한 도메인 분석 및 정의

$$j = \{ \sum_{i=1}^{n} (jn_l, jp_j, jsl_k, ir_l, jt_m, jd_P, jh_q, jps_l, jc_s) \}$$

1. 작업번호(Job Number, *jn*) : 고객으로부터 요청된 업무로 접수된 순서로 번호가 부여된다.
2. 작업요청부분(Job Part, *jp*) : 문제가 발생된 부분으로 하드웨어, 소프트웨어 또는 모두가 필요한지를 구분한다.
3. 작업 요구기술 단계(Job Skill Level, *jsl*) : 최소 어느 단계의 기술이 필요한지를 정의하며 1 에서 10단계로 구분된다.
4. 작업지역(Job Region, *jr*) : 어느 지역에 요청된 업무 인지를 구분하며 영업지역으로 구분한다.
5. 작업 소요시간(Job Duration, *jd*) : 작업을 마치는데 소요되는 시간으로 분으로 표현한다.
6. 작업 가능시간(Job Timing, *jt*) : 작업이 가능한 시간대를 설정한다.
7. 작업우선순위(Job Status, *js*) : 요청된 작업이 어떤 특정한 시간대를 요구하면 Timing으로 구분하며 최우선순위를 요구하면 Special, 보통의 요청이면 Normal로 구분된다.
8. 작업의 처리상태(Job Process Status, *jps*) : 요청된 작업의 처리상태로 Assign은 배정된 상태, Wait는 미배정 상태, Complete는 처리완료 상태로 구분된다.
9. 작업 유무상 여부(Job Cost, *jc*) : 접수된 작업이 무상인지 유상인지를 구분하며 작업배정의 우선순위를 정할 때 유상처리는 무상처리보다 우선한다.

2) 인력에 대한 도메인 분석 및 정의

1. 인력의 번호(Human Number, *hn*) : 업무에 배정할 수 있는 인력을 정의한다.
2. 인력의 기술부분(Human Part, *hp*) : 인력이 종사하는 부분을 정의한다.
3. 인력의 보유기술 단계(Human Skill Level, *hsl*) : 인력이 보유하고 있는 기술단계를 정의한다.
4. 인력의 영업지역(Regional Restriction, *hr*) : 인력이 주로 담당하고 있는 지역을 정의한다.
5. 인력의 상태(Human Status, *hs*) : 인력의 현재 상태로 작업이 배정된 상태는 Working로 근무가능하지만 작업이 배정되지 않는 상태는 Nonworking, 휴무중인 상태는Rest로 회의중인 상태의 인력은 Meeting으로 교육중인 상태에는 Education로 구분하여 정의한다.

6. 근무시작시간(Human Working Start Time, *hts*) : 업무 시작 시간을 설정한다.
7. 근무종료시간(Human Working End Time, *the*) : 업무 시작 시간을 설정한다.

3.3.2 제약조건 분석

1) 제약조건의 분류

(Ⅰ) 절대적 제약조건(Hard Constraint)

1. 강한 절대적 제약조건(Strong Hard Constraint)

이런 부류의 제약 조건은 사람이나 시스템에 의해서 절대로 위배되어서는 안 되는 조건으로 같은 시간에 한 사람이 서로 다른 두개의 작업을 수행 할 수 없다는 조건이 이에 해당된다.

2. 약한 절대적 제약조건(Weak Hard Constraint)

일상적인 상황에서는 절대로 위배할 수 없으나 어떤 특별한 상황에서는 위배할 수 있는 제약조건이다. 이런 종류의 제약조건은 시스템이 능동적으로 위배 할 수는 없지만 인간의 일시적인 지시가 있는 경우에는 시스템은 이를 수용하여 지시한 그대로 스케줄링 하여야 한다. 예를 들어 어떤 작업은 고객의 요청에 의해서 지정된 요원을 요청하거나 영업부서에서 동행을 요구하는 경우 또는 인력이 교육에 참가한 경우 이 인력에게 업무를 배정할 수 없으나 특별한 경우에는 이를 반영하여 스케줄링을 해야 한다.

(Ⅱ) 유연한 제약조건(Soft Constraint)

기본적인 제약조건에는 파견금지 지역이나 특별 인력에 대해서는 기본 조건을 위배하면서도 시스템이 능동적으로 스케줄 할 수 있는 제약조건이다. 일반적으로는 요구되어지는 기술레벨의 그 이상단계를 파견하지만 인력이 부족하면 요청되어진 기술레벨의 1단계정도의 하급 요원을 유연성 있게 파견한다거나 유휴인력이 있을 경우 고급단계의 요원과 동행하여 파견되거나 타지역으로 파견되어 질 수 있다.

2) 제약조건의 선언 및 정의

1. 일정단계의 기술을 요구하는 작업은 담당지역 내에서 처리해야 한다.
   IF $([jn_j, jsl] <= k)$ (단, $k \in jsl$)
   Then $[jn_i, jr] = [hn_j, hr]$
   (단, *jn* 은 작업번호, *jr*은 작업지역, *jsl*은 작업에 필요한 기술단계, *hn*은 인력, *hr*은 담당지역)

2. 요구된 작업의 기술단계보다 인력의 기술단계가 그 이상이어야 한다.
   $[jn_i, jsl] <= [hn_j, hsl]$
   (단, *jn* 은 작업번호, *jsl*은 작업에 필요한 기술단계, *hn*은 인력, *hsl*은 인력의 보유 기술단계)

3. 한 작업에 파견된 인력은 2명(H/W,S/W)이하이어야 한다.
   $[jn_i, Sum(hn_i)] <= 2$
   (단, *jn* 은 작업번호, *hn*은 인력)

4. 배정된 작업을 마친 후 다음장소로 이동하기 위해서는 최소 40분 이상이 필요하다. 이때의 이동시간은 같은 지역 내에서 배정된 장소에서의 이동시 적용된다. jn_1을 마친 후 jn_2의 작업을 시작하기 위해서는 다음과 같은 시간이 필요하다.
   IF $[jn_1, jr] = [jn_2, jr]$ *Then*

jn_1=(S_1, D_1)
jn_2=(S_2, D_2)
(단, S는 작업시작시간, D는 작업에 소요되는 시간)
$jn_i(S)$ ($jn_{i-1}(S_{i-1})$ + $jn_{i-1}(D_{i-1})$ + 40
(단, $jn_i \leq J$) 여기서 n은 작업의 수이다.

5. 요구된 작업량이 인력에 비해 많을 경우 일정단계 이상의 기술인력은 다른 영업지역에 파견 될 수 있다.
([jn_j ,JSL] >= k) AND ([jn_j ,jps] = {Wait}) AND (|[hn_i ,hs(Nonworking)]| != 0)
(단, $k \subset jsl$) (단, *jn* 은 작업번호, *jsl*은 작업에 필요한 기술단계, *jps*는 작업의 처리상태, *hn*은 인력, *hws*는 인력의 작업배정상태)

6. 시스템에 의한 자동배정금지 업무가 존재하는 경우는 인력의 상태가 회의, 교육, 휴가 중인 경우 시스템은 이 인력에게 작업을 배정 할 수 없다.
[h_s, h_s]= MEETING OR [h_s,h_s]= EDUCATION OR [h_s, h_s]= REST

7. 작업의 시작시간이 인력의 근무시작시간보다 30분 이상이 될 경우 작업을 배정하지 않는다.
(단, 작업이 요구시간대를 갖지 않으면)
[hn_i ,hts] > [jn_j ,S] + 30
(단, S는 작업시작시간, D는 작업에 소요되는 시간)

8. 작업의 종료시간이 인력의 근무종료시간보다 60분 이상이 될 경우를 작업을 배정하지 않는다.
(단, 작업이 요구시간대를 갖지 않으면)
[jn_j ,S] + [jn_j ,D] < [hn_i ,hte] + 60
(단, S는 작업시작시간, D는 작업에 소요되는 시간)

9. 인력의 작업시작시간이 오전 10시 이전인 경우 오후 12시부터 1시까지는 되도록 작업을 배정하지 않는다.
(단,작업이 요구시간대를 갖지 않으면)
IF [hn_j ,hts] > 600 Then
{[jn_j ,S] + [jn_j ,D] < 720} AND {[jn_j ,S] > 780}
(단, S는 작업의 시작시간, D는 작업에 소요되는 시간)

3.3.2 A/S 작업배정 스케줄링 방법

유지보수 작업배정 스케줄링에서는 최적의 해를 두개의 측면에서 볼 수 있다. 비용을 최소화하여 스케줄을 하는 방법과 고객의 만족도를 최대화하여 스케줄링 방법이다. 여기서, 고객만족지수란 최종소비자에게 A/S서비스 품질에 대해 해당제품과 관련된 서비스를 받아본 고객이 직접 평가한 만족수준의 정도를 모델링에 근거하여 측정, 계량화한 지표를 의미한다[김99]. 각각의 방법은 서로 상충되는 양상을 보이는데 비용을 최소화하여 스케줄을 할 경우에는 고객의 만족도가 떨어지고 다음날로 작업이 미루어지게 되지만 비용이 줄어들고 고객만족도를 최대화하면 비용이 증가하지만 다음날로 작업이 이월되지 않으므로 스케줄링이 합리적으로 세워진다. 본 논문에서는 최적의 해에 대한 기준으로 스케줄링시 비용을 최소화 하고 고객만족도를 최대화하는 방법으로 가능한 많은 작업을 처리 할 수 있도록 하는 것을 최적의 해로 정의했다.

(1) 비용적인 측면에서의 추정 함수

함수를 생성하는데 사용된 값들은 실제의 데이터를 근거로 하여 추정해 낸 것이다. 가중치를 이용하여 함수를 구한 것이 다음의 수식이다. 비용적인 측면에서 업무처리건수에 대한 가중치를 구한 결과는 2차 방정식으로 표현되고 이를 수식으로 나타내면 다음과 같다.

$$y=ax^2(a>0),(x\geq 0),(y\geq 0)\text{.....수식[1]}$$

업무처리 건수를 m이라 하고 데이터의 실제값을 대입하면 a값의 평균을 다음과 같은 식을 이용하여 얻는다.

$$\frac{1}{m}\sum_{i=1}^{m}a_i$$

(2) 고객만족도 측면에서의 추정 함수

고객만족도 측면에서 업무처리건수에 대한 가중치를 구한 결과는 1차방정식으로 표현되고 이를 수식으로 나타내면 다음과 같다.

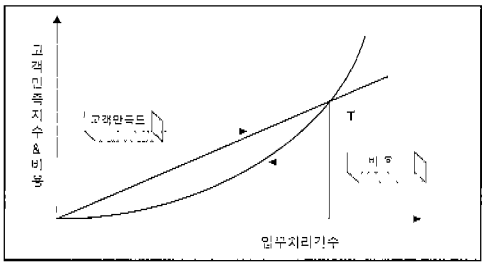
$$y=\beta x(\beta>0)\text{.....수식[2]}$$

업무처리건수를 m이라 하고 데이터의 실제값을 대입하면 β값의 평균을 다음과 식을 이용하여 얻는다.

$$\frac{1}{m}\sum_{i=1}^{m}\beta_i$$

(3) 최적화 추정 함수

작업처리건수와 처리해야 될 작업건수는 매일 변화하는 값이지만 전체적인 결과 값은 큰 영향을 미치지 못한다. 그림[5]는 각각의 추정함수를 나타낸 것으로 실제적으로 트래이오프(Trade-off)



〔그림 2〕 고객만족지수 및 비용에 대한 업무처리건수의 상관도표

는 두 함수가 만나는 점이다. 점 T 이상에서는 비용이 급격히 증가하지만 고객만족도가 완만히 증가하므로 요청되는 기술레벨에 맞게 파견되는 인력의 기술단계를 우선적으로 고려하여 스케줄링하고 점 T 이하에서는 비용은 완만히 증가하는 반면 고객만족도는 일정하게 증가하므로 파견되는 인력의 기술단계를 탄력적으로 적용하여 스케줄링을 해야 한다. 수식 [1]과 [2]를 적용하면 점 T 는 다음과 같이 구할 수 있다.

$$T=\frac{\beta}{a}$$

3.4 A/S 작업배정 스케줄링의CSP모델

A/S작업배정 스케줄링 시스템은 A/S작업배정 스케줄링에 필요한 기본데이터를 바탕으로 도메인을 분석하고 생성된 객체에 제시된 제약조건을 추가하여 스케줄링을 하게된다. 주어진 변수의 개수 또는 각 변수에 주어진 도메인의 크기가 증가할수록 해를 발견하는데 필요한 탐색공간 및 탐색시간이 증가하게 된다. 문제 해결 알고리즘을 적용하여 선언된 제약조건을 추가한 후 도메인 변수에 대하여 전향검사(Forward Checking)를 통하여 제약조건의 일치성을 검사하고 도메인 여과 기법을 적용하여 제약조건을 만족하는

```
IlcSchedule
DefineProblem( IlcManager m, IlcInt timeMax,
IlcInt JN,
IlcInt JD, Char** JS, IlcInt HN, IlcInt* HTS, IlcInt*
THE)
IlcSchedule schedule(m, 0, timeMax);
IlcInt i = 0;
For ( j = 0; j < JN; j++ ) {
Activity* previousActivity = 0;
IlcInt k;
For (k = 0; k < JD[i]; k++ )  {
Humans* human = GetHuman(HN[i]HSL[i],HR[i]);
PreviousActivity = new(m.getheap()) Activity(
```

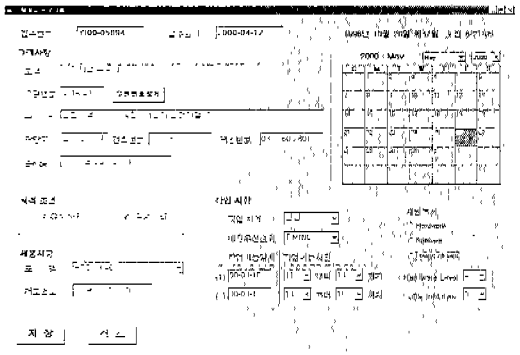
〔그림 3〕 ILOG를 표현한 문제 선언

인력을 찾는다. 만일 만족되는 인력이 없으면 그 다음 인력을 선택하여 위의 과정을 반복하게 된다. 만족되는 해를 찾으면 저장해 두었다가 스케줄링을 마친 후 결과를 출력하게 된다. 본 연구에서는 CSP의 라이브러리인 ILOG사의 ILOG Scheduler을 사용하여 스케줄링을 하였다. [그림3]은 ILOG로 A/S작업배정스케줄링의 문제를 표현한 것이다.

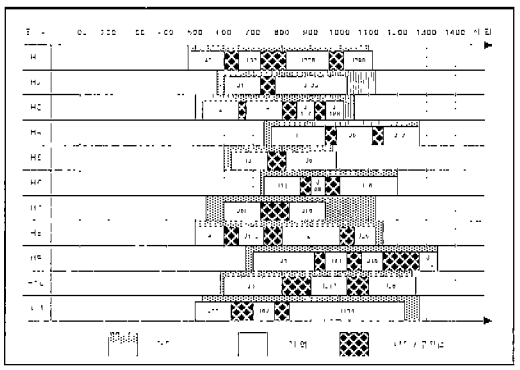
4. 실험 및 평가

본 연구에서 구현한 A/S작업배정 스케줄링 시스템은 Windows 98 환경에서 제약조건 프로그래밍(Constraint Programming)을 위한 유용한 함수들과 클래스들을 포함한 C++의 라이브러리 형태로 존재하는 ILOG Scheduler4.31을 이용하였다.

본 실험에서는 사용된 데이터는 무작위로 3일을 추출하였고 3일 동안 처리되어야 할 작업과 같은 날 인력의 근무시간 테이블을 사용하였다. 다음 그림[4]는 사용자의 데이터 입력 화면을 보여주



〔그림 4〕 A/S작업배정 스케줄링 시스템의 입력화면



〔그림 5〕 A/S작업배정 스케줄링 시스템으로 스케줄링한 결과

고 있으며 [그림5]는 스케줄링 된 결과를 보여주고 있다.

본 실험에서는 먼저 최적화 함수의 검증을 위하여 동일한 작업 건수인 781개를 가지고 최적화 함수를 적용한 스케줄링 방법과 적용하지 않은 방법으로 실험을 하였다. 그 결과는 [표4]와 같이 많은 차이점을 보여주고 있다.

최적화 함수를 고려하지 않은 경우 하루에 많은 작업을 처리할 수 있고 고객만족지수는 높아지지만 작업 당 소요비용이 급격히 증가한 결과를 보여주고 있다. 여기서 고객만족지수가 최적화 함수를 적용하지 않은 스케줄링이 높아진 이유는 작업에 대한 처리응답시간이 짧기 때문이

〔표 4〕 최적화 함수를 적용한 스케줄과 적용하지 않은 스케줄의 결과 비교표

항 목	최적화함수를 적용한 스케줄링	최적화함수를 적용하지 않은 스케줄링
처리된 작업의 수	781건	781건
작업처리에 소요된 인건비	37,700,000원	44,600,000원
작업처리에 투입된 인원 (투입된 인원/가용인원)	(97/116)명	(116/116)명
작업 당 평균소요비용	48,300원	57,100원
고객만족지수	83%	96%

고 고객만족지수는 서비스를 받은 고객에게 작업처리결과에 대한 전화조사로 측정이 되며 만족지수에 영향을 주는 변수 중 처리응답시간도 많은 영향을 미치기 때문이다.

[그림2]에서처럼 트래이드오프를 기준으로 고객만족지수(Customer Satisfaction Index)를 더 높이기 위해서는 작업처리건수를 증가시켜야 되지만 기술인건비용은 이 보다 더 크게 증가하고 있다. 이 결과는 앞에서 언급한 최적화 함수가 검증됨을 확인해 주고 있다.

[표5]의 결과는 3일 동안 접수된 작업의 수 941개, 투입된 인력의 수 120명을 가지고 최적화 함수를 적용하여 인력에게 작업을 배정하는 실험을 하였다. 위 와 같은 결과에서 시스템에 의한 스케줄링은 전문가에 의해서 스케줄링된 것 보다 적은 비용으로 많은 작업을 할 수 있었고 한 작업에 소용된 인건비의 감소는 업무의 효율성이 좋아졌다고 볼 수 있다. 또한, 인력의 갑작스러운 결근으로 인하여 재스케줄링을 요구할 때 언제든지 합리적인 재스케줄링이 가능했고 최적화 함수를 적용하여 고객만족지수의 수치도 높은 결과치를 보여주고 있다.

〔표 5〕 전문가와 시스템에 의해 스케줄링한 결과 비교표

항 목	전문가에 의한 스케줄링 결과	시스템에 의한 스케줄링 결과
처리된 업무건수	743건	781건
작업처리에 소요된 인건비	37,900,000원	37,700,000원
작업당 평균소요비용	51,300원	48,300원
스케줄링의 최적화	고려하지 않음	최적화 함수적용
재스케줄링 가능성	가능하지만 비합리적	합리적인 재스케줄링
고 객 만 족 지 수	79%	83%

5. 결론 및 향후 연구

A/S작업배정을 스케줄링하기 위해서는 많은 노력과 인력이 소요 되었고 인간의 실수로 인한 불합리한 스케줄링이 되었다. 또한, 작업처리 건수에 따른 고객만족 및 비용에 대해 고려를 할 수가 없었다. 그러나, 본 연구에서는 CSP기법을 스케줄링 문제에 적용한 A/S작업배정의 스케줄링 시스템에 의해 자동적으로 작업을 배정함으로써 실수로 인한 불합리한 스케줄링을 배제 할 수 있었고 인력에게 비효율적인 업무 배분으로 많은 처리비용이 들었지만 최적화 함수를 적용하여 스케줄링을 함으로써 적은 비용으로 많은 작업을 처리 할 수 있었을 뿐만 아니라 부수적으로 조직내에 고용된 인력에게는 편협적인 작업배정으로 인한 인력들의 수당에 대한 불만사항을 해결 할 수 있었고 인력의 수급상황을 고려하여 인력관리에도 많은 도움을 줄 수 있게 될 것이다. 또한, 인력의 갑작스러운 부재로 인한 재스케줄링에 대한 문제도 해결 할 수가 있었으며 무엇보다 중요한 서비스 사용자에게 서비스에 대한 만족을 충족시켜 줄 수 있었다.

향후, 본 연구에서는 사용자, 즉 고객의 측면에서 서비스를 네트워크상에서 신청하고 시스템이

자동적으로 사용자의 요구를 수용하여 좀 더 진보적인 A/S작업배정 스케줄링을 하는 에이전트(Agent)기반의 대화형시스템으로 연구를 모색하고자 하며 GPS(Global Positioning System)를 이용한 최적의 이동경로를 고려한 스케줄링에 대한 연구도 필요하다. 이는 인터넷으로 전자제품의 수리서비스를 신청하고 시스템은 즉각적인 방문일정과 인력을 보여줌으로써 서비스공급자의 신뢰성이 높임으로써 한 단계 높은 고객만족을 줄 수 있는 시스템을 개발할 수 있을 것이다.

참 고 문 헌

[1] 양종윤, "스케줄링 문제해결을 위한 지식 기반기법과 제약만족 기법의 비교 연구", 인하대 석사 논문 1998.2

[2] [Jo 97] Jo Geun-Sik, Jong-Jin Jung, Chang-Yoon Yang, *"Expert System for Scheduling in an Airline Gate Allocation"*, Expert System International Journal, Elsevier Science Ltd, Nov, 275 (282,1997.

[3] [Heintze 92] N.Heintze, J.Jaffar, S.Michayov, P.Stucky and R.Yap, *"The CLR(R) Programmer's Manual Version 1.2", IBM T. J.. Watson*, 1992.

[4] [Dhar 90] Dhar. Vasant, Nicky Ranganathan, *"Intger Programming vs. Expert Systems : An Experimental Comparison"*, ACM, Vol.33, No.3, 323(336, 1990.

[5] [Pascal 89] Pascal Van Hentenryck, *"Constraint Satisfaction in Logic Programming"*, MIT Press, U.S.A, 1989.

[6] [Mackworth 97] A. K. Mackworth, *"Consistency in Networks of Relations"*, American Association for Artificial Intelligence, pp.99-118, 1997.

[7] [Frost 94] D. Frost, R. Dechter, *"In search of the best constraint satisfaction search"*, Artificial Intelligence, pp.301-306, 1994.

[8] [Wallace 94] R. J. Wallace, *"Why AC-3 is Almost always Better than AC-4 Establishing Arc Consistency in CSPs"*, Artificial Intelligence, pp.239-345, 1994.

[9] [Nadel 89] Nadel B.A., "Constraint *Satisfaction Algorithm"*, Computational intelligence 5, 1989.

[10] [Monte94] Montc Zweben, Mark S. Fox *"Intelligent Scheduling"*, Morgan Kaufmann Publisher, 1994.

[11] [Philippe 95] Philippe Chretinne, Edward G. Coffman. Jr, Jan Karel Lenstra, Zhen Liu, *"Scheduling Theory and its Applications"*, JOHN WILEY & SONS, 1995

[12] [Fox87] Mark S. Fox, *Constraint-Directed Search : A Case Study of Job-Shop Scheduling*, Research Notes in Artificial Intelligence, Morgan Kaufmann Publishers, Inc.1987 http://kti.ms.mff.cuni.cz/~bartak/constraints/intro.html

[13] [김99] 김건우, "고객만족과 기업성과의 관계", 한국고객만족확회 춘계학술대회 Vol1.No1, April 1999.