웹 환경 하에서의 제약 만족 기법에 의한 공간 계획 시스템

(A Web-based Spatial Layout Planning System with Constraint Satisfaction Problems)

정 재 은 † 전 승 범 †† 조 근 식 †††

(Jae-Eun Jung) (Seung-Bum Jeon) (Geun-Sik Jo)

요 약 공간 계획 시스템(Spatial Layout Planning System)은 사용자의 요구에 따라 사각의 자원을 일정 공간 안에 할당하고 사용자의 만족도를 최대화함으로써 공간 효율성을 최적화하는 시스템이다. 공간 계획 문제는 방대한 범위의 공간을 탐색해야 하므로 시간과 공간적 측면에서 높은 복잡도(Complexity)를 갖는 문제이다. 또한 특정 영역의 수정 요구나 재설계 요구와 같은 사용자의 동적인 요구 사항들을 수용할 수도 있어야 한다. 본 논문에서는 CSP(Constraint Satisfaction Problems) 해결 기법 기반의 자원 할당법을 이용함으로써 효과적으로 공간 계획 문제를 해결할 수 있도록 하였으며 사용자의 요구에 따라 변화되는 제약조건은 지능형 사용자 인터페이스 모델을 통해 좀 더 향상된 결과가 도출될 수 있도록 설계 및 구현하였다. 또한, 2차원 도면에서의 수정 요구에 대한 편이성과 시각적 검증을 위해 웹 환경 하에서의 VRML(Virtual Reality Modeling Language)을 이용한 3차원 도면을 보여준다.

Abstract The spatial layout planning system allocates rectangular resources in the limited space according to user requirements. This system also can optimizes the spatial allocation problem to maximize the user's requirement. The spatial layout planning problems for this system can be solved by searching a wide area of space since this problem entails the non-polynomial algorithm. By accommodating the user's dynamic requirements, the modification of a specific space and the redesign of the whole area can be accomplished. In this paper, the spatial layout planning problem is solved efficiently with a resource allocation method based on CSP. The dynamic constraints by adding user requirements are accommodated through the intelligent user interface. The 3-D layout on the web environment by using VRML is also shown for providing for the visual verification of the 2-D layout and, thereafter, the additional modification of the 2-D layout.

1. 서 론

공간 계획(Spatial Layout Planning)이란 2차원 형태의 자원(2-Dimensional Resource)들을 입력받아서 주어진 2차원 공간 내에 이들을 배치하는 할당 문제(Allocation Problem)임과 동시에 공간의 활용 효율성을 최대로 하는 최적화 문제이다. 이러한 공간 계획 문제는 일반적으로 2차원 공간이 넓을수록, 자원의 수가 많을수록 탐색공간과 계산시간을 지수적으로 증가하게 하는 경향이 있다. $x \times y$의 공간에 n개의 직사각형 자원을 할당한다고 하면 $O(x \cdot y^{2n})$의 복잡도를 가지며, 이것은 NP-complete 문제라고 할 수 있다 [1].

공간 계획의 대표적인 문제 중 하나인 주택 공간 계획은 제한된 주택 공간 내에 마루, 침실, 부엌 등의 직사각형의 형태를 띄고 있는 자원들을 사용자가 원하는 조건들뿐만 아니라 각각의 자원들 사이의 관계(Relationship)에 따라 배치하는 작업이다. 이는 공간 계획 문제에서 나타나는 대부분의 어려움을 모두 가지고

† 비 회 원 : 인하대학교 전자계산공학과
jungjeff@orgio.net
†† 비 회 원 : 거림시스템(주) 거림연구소 연구원
††† 종신회원 : 인하대학교 전자계산공학과 교수
gsjo@dragon.inha.ac.kr
논문접수 : 1999년 7월 6일
심사완료 : 2000년 1월 27일

있으며, 특히 최적화 문제의 측면에서 볼 때, 개인적 성향에 따라 상당한 차이를 보이게 되는 사용자 만족도를 정량적으로 나타낼 수 있는 최적 함수를 정의하기 힘들다.

위의 문제들을 해결하기 위해 과거에도 많은 연구가 이루어져왔다. 코펜하겐 대학에서 Prolog를 이용하여 2차원 공간을 활용하기 위한 Floor Plan Design 문제에 대해 지식 기반 시스템(Knowledge-Based System)을 이용한 방법론이 제시되었다 [2]. 그리고 일본의 도쿄가스 공사에서는 CBP(Constraint Based Planner)라는 CLP(Constraint Logic Programming) 기법을 이용한 주택 공간 설계 시스템을 개발하였다 [3]. 하지만 이러한 공간 계획 시스템들은 공간 계획 문제가 가지고 있는 높은 복잡도에 의해 많은 시간이 걸릴 뿐만 아니라 사용자의 요구를 만족시킬 수 있을만한 결과를 만들어 내지 못한다. 이러한 환경 하에서 최근 인공지능의 새로운 분야로 활발히 연구가 진행되고 있는 CSP 해결 기법을 이용함으로써 보다 효율적인 문제 해결이 가능하게 되었다 [1].

본 논문에서는 기존의 공간 계획 분야에 대해 연구되었던 기술들을 소개하고 문제점을 파악하여 그 해결 방법으로 CSP 해결 기법을 제안한다. 또한 사용자의 동적인 요구 조건을 충족시킬 수 있는 지능형 인터페이스 시스템(Intelligent Interface System)을 설계하였다. 그럼으로써, 사용자에게 3차원 설계도면이 제공되어 직관적으로 인식할 수 있도록 하였다. 이와 더불어 브라우저를 통해 웹 환경 하에서 사용자의 간접 체험을 경험할 수 있도록 VRML(Virtual Reality Modeling Language)을 이용한 가상 현실 시스템을 구현하였다.

2. 공간 계획 문제의 접근 방법 및 CSP 해결 기법

2.1 정수 프로그래밍(Integer Programming)

정수 프로그래밍은 제약 조건을 기호적 제약 조건과 수치적 제약 조건으로 모델링한다. 정수 프로그래밍에서는 기호적 제약 조건을 표현하기 위해서 대수 연산자(Algebraic Variable)로 연결된 부울 변수(Boolean Variable)를 사용한다. 하지만 정수 프로그래밍 형식은 문제의 규모가 커지면 커질수록 방대한 양의 변수들을 필요로 할뿐만 아니라, 이들 사이의 복잡한 상관 관계(relation)를 요구하게 된다. 따라서, 이러한 공간 계획 문제를 정수 프로그래밍으로 모델링하는 것은 복잡하고 어려운 과정을 거쳐야 한다. 특히, 모델링 할 때 제약 조건의 논리곱(Conjunction)관계는 표현하기가 용이하나 논리합(Disjunction)관계의 경우에는 표현과 수식화(Formulation)가 어렵고, 또한 기호적 제약 조건을 정수 프로그래밍으로 해결하기 위해서는 많은 양의 인위 변수(Dummy Variable)를 필요로 한다 [4]. 뿐만 아니라 이러한 복잡한 모델링 작업은 동적인 사용자의 요구 변화에 따른 시스템에 대한 수정을 어렵게 한다는 한계를 내포하고 있다.

2.2 지식 기반 시스템(Knowledge-based System)

지식기반 해결 방법은 주어진 문제를 논리적으로 자연스럽게 표현할 수 있으며 기호적인 제약 조건의 처리가 편하다. 즉, 지식 표현(Knowledge Representation)이 쉽다. 따라서 실세계에서 발생되는 다양한 문제 해결 분야, 특히 사례 기반 추론(Case-based Reasoning)과 같은 문제 해결 분야에서 널리 사용되어 왔다. 그러나 논리 언어는 수치 제약 조건의 처리에 대한 어려움 때문에 기호적인 제약 조건의 처리와 함께 수치 제약 조건의 처리도 동시에 요구되는 문제의 적용에 많은 한계를 나타내고 있다. 또한 지식 표현의 일관성(Consistency) 유지가 제대로 되지 않을 경우 시스템의 확장이나 수정 시 상당한 노력이 요구되어지며, 도메인 여과(Domain Filtering) 과정에 의해 해를 구하는 도중에 탐색 공간(Search Space)을 축소하지 못하기 때문에 문제를 해결하는데 있어서 필요 이상의 시간이 소요된다 [5].

2.3 CSP(Constraint Satisfaction Problems) 해결 기법

공간 계획 문제는 그 문제의 특성상 CSP로 적용하기 쉽고 이러한 접근은 기존의 지식 기반 시스템의 접근보다 속도적인 측면에서 효과적이라 볼 수 있다. CSP는 여러 가지 제약 조건을 만족하는 해를 구하는 것으로 CSP의 구성요소는 변수, 도메인, 그리고 변수들간의 제약 조건으로 이루어진다. CSP는 변수의 개수와 각 변수에 들어갈 도메인의 범위에 의해 지수적으로 증가하는 복잡도를 가지고 있으므로 큰 변수나 도메인의 경우에는 많은 시간이 걸린다. 이러한 문제를 해결하기 위해 각 변수들을 전향검사(Forward Checking)를 이용하여 제약 조건의 일치성 검사(Consistency Checking)을 하여 도메인 여과(Domain Filtering)를 적용하면 전체 탐색 공간의 크기를 현저히 줄일 수 있다 [6][7][8].

3. 웹 환경 하에서의 주택 공간 계획 시스템

3.1 주택 공간 계획 시스템의 구성

본 논문에서 구현된 시스템은 [그림 1]과 같이 웹 환경 하에서 구현된다.

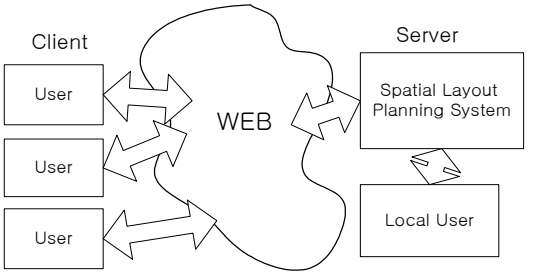


그림 1 웹 환경 하에서의 시스템 구성

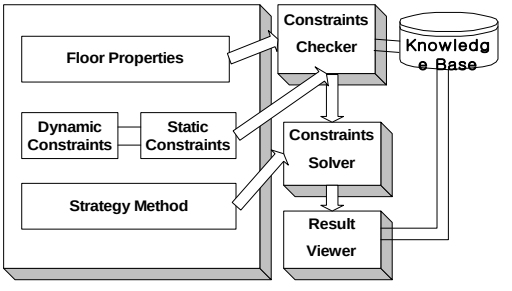


그림 2 주택 공간 계획 시스템 내부 모듈의 구성

웹 환경 하에서의 사용자와 인터페이스를 위해 클라이언트-서버 구조를 이용한다. 서버는 사용자들에게 VRML(Virtual Reality Modeling Language) 코드를 보내 줌으로써 사용자들은 주택 공간 계획의 결과를 인터넷 브라우저를 통해 볼 수 있게 된다.

주택 공간 계획 시스템의 내부 모듈 구성은 [그림 2]와 같다. 모듈 각각은 클라이언트 측으로부터 사용자의 요구 조건을 입력받는 데이타 입력 모듈, 사용자의 요구사항의 오류 등의 제약 조건의 충돌여부를 검증하는 제약 조건 검증(Constraint Checker) 모듈, 최종 결과 도출을 위한 문제 해결(Constraints Solver) 모듈, 로컬 사용자에게 결과를 보여주기 위한 2차원 결과 출력(Result Viewer) 모듈로 구성된다.

본 시스템의 수행과정은 [그림 3]과 같다. 사용자는 우선 자신들이 원하는 공간에 대한 속성, 각 방들에 대한 속성 및 사용자 요구사항을 입력한다. 이렇게 입력한 데이타는 제약 조건 검증 모듈로 들어가 제약 조건의 정당성을 검증 받게된다. 이때, 제약 조건의 충돌로 인한 오류 발생 시 수정 메시지를 사용자에게 보여주고 재입력을 요구한다.

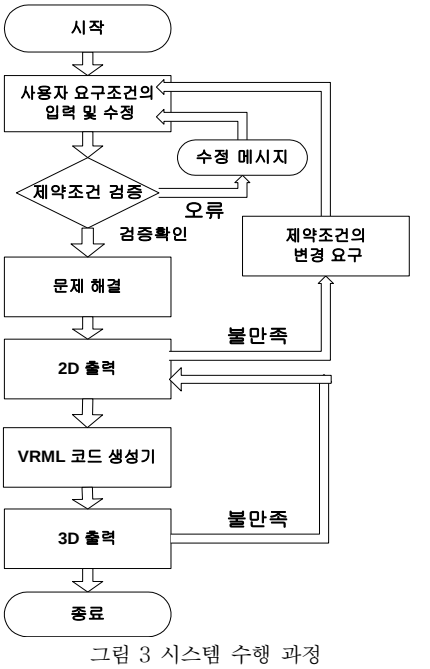


그림 3 시스템 수행 과정

제약 조건 검증 모듈을 통과한 데이타들은 문제 해결 모듈로 들어가게 되며 해를 구하는 작업에 돌입한다. 우선 해를 구하게 되면 2차원 출력을 하게 되고 사용자의 불만이 없을 경우에 마지막으로 3차원 설계도면 또는 VRML 코드를 생성하게 된다.

3.2 공간 계획 문제

3.2.1 자원 할당법

제한되어 있는 주택공간 내에 방을 배정하기 위한 방법으로는 2차원 공간 내에 수직선과 수평선을 그어 가면서 분할해 가는 공간 분할법(Space Partitioning Method)과 2차원 평면을 끼워 맞춰가면서 할당하는 자원 할당법(Resource Allocation Method)등이 있다.

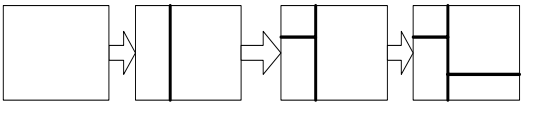


그림 4 공간 분할법

공간 분할법은 [그림 4]와 같이 하나의 사각 공간을 수직선과 수평선으로 금긋는 작업을 재귀적으로 수행하는 방법이다. 공간이 자원의 수만큼 나눠지면 각각의 자

원을 배치한다. 이렇게 배치된 결과가 제약조건을 만족하는지 검사하여 만족하지 않으면 백트랙킹하여 만족하는 해가 나올 때까지 위의 작업을 반복 수행하게 된다 [2]. 따라서 이 방법으로는 나눠진 어떤 공간이 자원을 대응시킬 공간인지 아니면 다시 나눠질 공간인지 판단이 불가능하다.

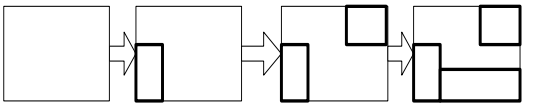


그림 5 자원 할당법

반면에 자원 할당법은 [그림 5]와 같이 2차원의 공간 안에 배치될 후보 자원들을 선택한 후, 그 자원이 갖는 제약조건을 검사한다. 제약 조건을 만족하게 되면 공간 내에 배치 가능한 위치에 고정시키게 된다. 이 작업을 모든 자원이 배치 가능할 때까지 반복하여 해를 구하게 된다 [9]. 그러므로 자원과 공간사이의 구분이 확실하다. 또한 CSP의 구성 요소인 변수, 도메인, 제약 조건들 사이의 관계가 자원들, 자원이 위치할 공간, 그리고 자원들간의 제약 조건들 사이의 관계와 유사하여 본 논문에서는 자원 할당법을 사용하였다.

```
procedure Resource Allocation Algorithm
begin
do
  Select a resource
  if  There is a space of constraints satisfying enough to
      allocate the resource in field
  then  Allocate the resource and propagate constraints of
        the resource
       for reducing domain values of another resources
  else
  if  There is another space enough to allocate the resource
      in field
  then  Allocate the resource and set propagate constraints
        for reducing another domain values
  else do
  if  There is a resource that can be removed.
  then  Remove the resource and backtracking
  else  Allocation planning is failed
  while  All resources are allocable
  Allocation planning is succeeded
end.
```

그림 6 자원 할당 알고리즘

3.2.2 자원 할당법을 이용한 문제 해결 알고리즘

제약 조건 검증을 마친 후 각 방의 속성 및 사용자 요구 조건은 문제 해결 모듈로 전달되어 해를 구하는 작업이 수행된다.

[그림 6]의 자원 할당법에 기반 하여 문제를 해결하게 된다. 우선 사용자가 입력한 자원과 제약 조건을 바탕으로 하나의 자원에 가능한 값을 지정한 후 그 자원이 공간 내에 위치할 수 있는지, 그리고 사용자 요구 조건에 적합한지를 검사한 후 공간에 방을 할당한다. 만약 할당된 방과 관련된 제약 조건을 전파(Propagation) 시켜 다른 방의 할당 가능 공간이 존재하지 않을 경우는 백트랙킹이 일어난다. 모든 방이 공간 내에 위치 가능할 때까지 이 과정을 반복한다.

3.2.3 휴리스틱(Heuristic)에 의한 할당 우선 순위 배정

일반적으로 자원 할당 문제에서는 자원을 할당하는 순서가 비교적 상당히 중요한 요인으로 작용할 수 있다.

표 1 자원 할당 순서에 대한 휴리스틱의 종류

자원의 크기에 따른 배정 순서	크기가 가장 큰 방부터 배정
	크기가 가장 작은 방부터 배정
자원의 할당 범위에 따른 배정 순서	크기 범위가 큰 방부터 배정
	크기 범위가 작은 방으로 배정

자원 할당 과정이 제약 조건 전파(Constraint Propagation)에 따른 도메인 여과 작업에서 현저한 차이를 보이기 때문이다 [10]. 자원 할당 순서에 대한 휴리스틱의 종류는 [표 1]와 같다.

전향 검사를 통한 도메인 여과 기법의 사용 시 일반적으로 도메인 축소효과가 높은 것을 먼저 할당하는 것이 백트랙킹의 횟수를 줄여 해를 찾는데 걸리는 시간을 효율적으로 줄일 수 있다 [7][10]. 그래서 본 시스템에서는 방의 넓이가 큰 것에 우선 순위를 높게 줌으로써 먼저 배치할 수 있게 하여 다른 방의 배정공간을 최대한으로 줄일 수 있게 하였다. 그리고 방의 크기 범위가 크게 되면 하나의 방에서 여러 모양이 나올 확률이 커지므로 크기 범위가 작은 방일수록 우선 순위가 커지도록 하였다.

3.3 CSP에 의한 문제 해결 과정

본 시스템에서는 각 자원들의 위치, 크기, 방향과 같은 속성 및 자원들 사이의 관계에 따른 제약조건의 만족 여부, 그리고 사용자 요구사항과 휴리스틱의 설정 등을 통해 다음의 문제 해결 과정을 수행하게 된다.

3.3.1 도메인 생성

각 자원의 속성과 데이타를 입력한다.

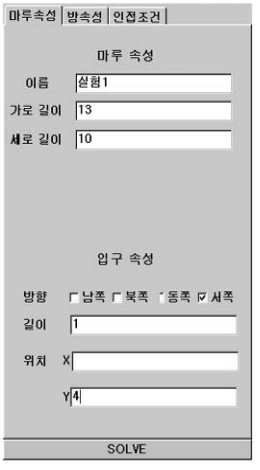


그림 7 마루 속성의 입력 화면

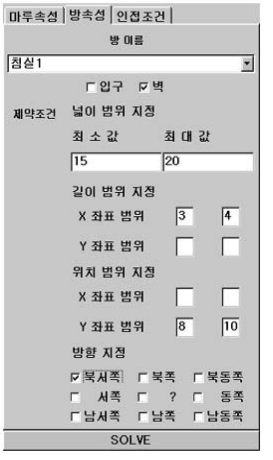


그림 8 방의 속성 입력 화면

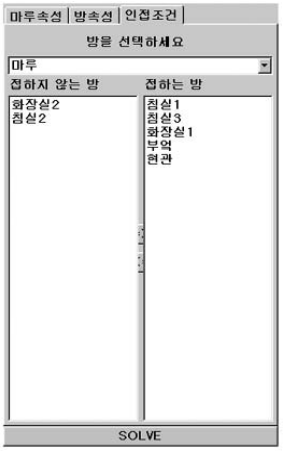


그림 9 인접 제약 조건의 설정 화면

[그림 7]과 같이 주택 공간의 이름, 가로 및 세로의 크기와 현관의 방향, 위치, 크기를 지정한다. [표 2]를 참조하여 침실, 화장실과 같이 주위가 벽으로 둘러 쌓인 형태, 마루, 부엌 같은 사방으로 트여있는 형태, 출입구의 위치나 크기 등에 제한을 받게 되는 현관같이 특수한 방 등의 속성을 지정하게 되는데, 각 방들의 크기, 위치, 길이 제한은 [그림 8]과 같고, 방들끼리의 인접사항에 대한 사용자 요구조건은 [그림 9]와 같다.

표 2 방 속성에 따른 구분

	벽으로 둘러 쌓임	위치 제약 조건의 유무		크기 제약 조건의 유무	인접 제약 조건
		하수도	채광		
안방	o	x	o	o	마루
침실	o	x	o	o	마루, 화장실
화장실	o	o	x	o	마루, 화장실
마루	x	x	o	o	안방, 침실, 화장실, 부엌
부엌	x	o	o	o	마루

3.3.2 제약 조건의 종류 및 표현

(1) 제약 조건의 종류

각각의 속성에 연관되어 있는 제약 조건은 기하학적 타당성을 위한 정적 제약 조건과 사용자의 요구 조건을 수용하기 위한 동적 제약 조건의 두 종류로 나뉜다.

1) 정적 제약 조건

i) 방의 위치와 길이는 정수 값을 갖는다. (데이타 타입에 대한 제약 조건)

- CSP는 유한 이산 도메인(finite discrete domain)에 대한 문제 해결 능력을 갖고 있기 때문에 방의 위치 및 크기에 해당하는 4개의 변수에 할당되는 값은 정수의 형태를 가진다. 하지만 실제 도면상에서의 실수 영역을 수용하기 위해서는 단위 길이를 상대적으로 줄여 나가는 방법이 있다. 예를 들어, 0.1 미터는 10 센티미터로 표현함으로써 단위 길이를 100배 축소하여 문제를 해결할 수 있다.

ii) 방의 넓이의 합계는 필드의 크기와 같다.

- 주택공간 내에 방이 배치될 때, 빈 공간이 존재하면 안 된다.

iii) 방의 위치는 필드를 벗어날 수 없다.

- 주택공간 내에 방이 배치될 때 주택공간을 벗어나는 경우는 금한다.

iv) 각각의 방들은 서로 겹치지 않는다.

- 각각의 방들은 서로 일정한 두께의 벽을 사이에 두고 위치한다.

2) 동적 제약 조건

i) 크기 범위의 한정

- 방의 가로 길이와 세로 길이의 곱은 사용자가 지정한 크기 범위 이내여야 한다.

ii) 길이 범위의 한정

- 방의 가로 길이와 세로 길이의 최소값 및 최대값을

정방형에 가깝거나 혹은 한쪽으로 긴 형태, 또는 방의 모양을 고정시키고 싶을 경우에 사용한다.

iii) 위치 범위의 한정

- 방의 배정 가능한 범위를 사용자가 원하는 범위 내로 축소시킨다.

iv) 가장자리 위치 설정

- 안방 혹은 마루의 경우와 같이 채광이 필요한 경우, 혹은 베란다와 같이 바깥 벽면에 위치해야 하는 경우 주택공간의 가장자리에 방을 위치시킴으로써 이를 만족할 수 있게 한다.

v) 인접 설정

- 어떤 방에서 다른 방으로 직접 이동을 할 수 있어야 할 경우(마루와 안방의 경우 등)에 부여하게 되는 제약 조건이다. 두 개의 방이 공통의 문을 위치시킬 수 있도록 서로 겹치고 있는 벽을 가지고 있어야 한다.

vi) 특별한 속성에 따른 방들의 설정

- 현관의 경우와 같이 현관의 위치와 크기에 따라 일반적인 방의 제약 조건보다 강한 제약 조건이 붙게 되는 경우이다. 방의 속성에 따라 위의 제약 조건들이 적절히 융합된 형태로 표현된다.

(2) 제약 조건의 표현

본 시스템에 사용되는 자원은 [그림 10]과 같이 좌변의 x좌표, 상단의 y좌표, 가로 길이, 그리고 세로 길이의 4개 속성이 제약 조건인 객체이다. 좌변의 x좌표와 상단의 y좌표가 갖는 도메인 값의 범위는 그 방의 위치에 대한 제약 조건이며, 가로 길이와 세로 길이가 갖는 도메인 값의 범위는 그 방 의 크기에 대한 제약 조건이다.

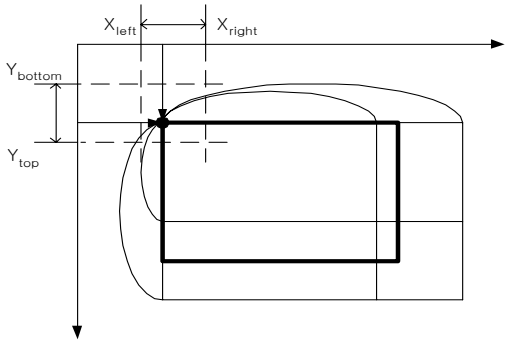


그림 10 자원의 제약 조건 표현

```
// 두 개의 방 어느 한 변은 동일 선상에 위치한다.
(location_X[a]+size_X[a]==location_X[b]) + (location_Y[a]
 +size_Y[a]==location_Y[b])
+(location_X[b]+size_X[b]==location_X[a]) + (location_Y[b]
 +size_Y[b]==location_Y[a])
 == 1
//만약 세로변이 일치할 경우 일치하는 선분의 길이는 d 이상을 가
 진다.
IF (location_X[a]+size_X[a]==location_X[b])
  || (location_X[b]+size_X[b]==location_X[a])
THEN (location_Y[a]+size_Y[a]-location_Y[b]>=1)
  && (location_Y[b]+size_Y[b]-location_Y[a]>=d)
//만약 가로변이 일치할 경우 일치하는 선분의 길이는 d 이상을 가
 진다.
IF (location_Y[a]+size_Y[a]==location_Y[b])
  || (location_Y[b]+size_Y[b]==location_Y[a])
THEN (location_X[a]+size_X[a]-location_X[b]>=1)
  && (location_X[b]+size_X[b]-location_X[a]>=d)
 location_X[n] : n번째 방의 좌측 x좌표  location_Y[n] : n번째
 방의 좌측 y좌표
 size_X[n] : n번째 방의 가로 길이  size_Y[n] : n번째 방의 세로
 길이
 a, b : 방 번호      d : 문 너비
```

그림 11 인접 제약 조건의 표현

[그림 11]은 두 개 이상의 자원들 사이에 존재하는 인접 관계 설정에 관한 제약 조건을 나타낸다 [1].

- 두 개 방의 어느 한 변은 동일 선상에 위치한다.
- 각 방의 접하고 있는 벽의 길이는 d 이상이다. (d는 접하고 있는 벽의 문 너비)

3.3.3 제약 조건 검증

사용자가 입력한 각각의 제약조건은 지식 기반의 제약조건검증 모듈을 통해 그 적합성을 검증 받게 된다. 사용자 요구사항의 입력이 완료된 후 문제해결을 시작하기 직전에 제약조건의 충돌 여부를 검증하여 적합치 않을 경우 사용자에게 에러 메시지를 나타내어 주어 불필요하거나 불가능한 해를 찾는데 소요되는 시간을 줄이는 역할을 한다.

지식 기반에서 검증하는 제약 조건 충돌 여부의 내용은 다음과 같다.

1) 방의 최소 넓이는 1 이상이며 마루의 넓이 이하이다.
2) 방들의 최소 넓이의 합은 마루의 넓이 이하이다.
3) 방들의 최대 넓이는 합은 마루의 넓이 이상이다.
4) 현관 속성을 나타내는 방은 오직 하나 존재한다.
5) 방의 최소길이는 마루의 한 면의 길이 이하로 제한한다.

6) 방의 최소 및 최대 길이의 범위는 방의 크기 제한에서 가능한 길이를 포함된다.

7) 방의 위치 범위 제한과 방의 최소 길이 제한의 합은 주택공간의 각각의 길이 이하 로 제한한다.

3.3.4 제약 조건의 동적 수용

문제의 특성상 최적해(Optimal Solution)는 사용자 개개인의 기호(Preference)에 따른 만족도가 가장 높은 해라고 정의할 수 있다. 그러나, 개개인마다 다른 기호를 모델링하기가 불가능하며 최적화 함수를 구하기도 힘들다. 사용자의 만족도는 대부분 자신들의 요구 사항이 얼마나 잘 수용이 되었나에 따라 결정되는데, 여러 가지 다른 요소들에 의해 달라질 수도 있다. 따라서 본 시스템에서는 사용자의 만족도를 향상시키는 방법으로 시행착오 방식(Trial-and-Error)을 취하고 있다. 즉, 사용자에게 최선이라고 추정되는 해를 제공하고 사용자에게 제약조건의 수정을 요구한다. 만약 사용자의 요구사항을 만족하는 해가 많을 경우나 해가 존재하지 않을 경우, 혹은 오히려 사용자의 만족도를 떨어지게 하는 해가 나올 경우 사용자와 시스템간의 상호작용에 의한 요구사항 변경이 필요하다. 이때, 제약 조건의 동적인 추가, 강화, 제거, 완화의 방법을 사용하여 해결할 수 있다.

(1) 제약 조건의 추가 및 강화

자원의 수가 적거나 자원들 사이의 제약이 느슨하여 사용자가 원치 않는 많은 해가 도출될 경우, 제약 조건의 추가 및 강화로써 이를 해결할 수 있다.

(2) 제약 조건의 삭제 및 약화

자원의 수가 많거나 자원들 사이의 관계가 너무 복잡하여 해가 나오지 않을 경우 제약 조건을 삭제하거나 완화시킴으로써 해를 구할 수 있다.

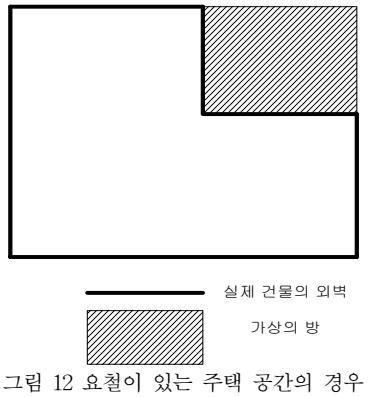


그림 12 요철이 있는 주택 공간의 경우

3.3.5 특수한 경우의 제약 조건 부여

1) 주택공간의 모양이 직사각형의 모양을 띄고 있지 않을 경우

[그림 12]와 같이 요철이 있는 주택공간에 방을 배치해야 할 때는 요철을 모두 포함할 수 있을만한 크기의 주택공간을 설정한 후 빈 공간을 채울 수 있는 방을 추가하여 위치 및 크기 범위의 한정 제약조건을 부여하는 식으로 해결이 가능하다.

2) 주택의 천장을 통한 채광이 가능할 경우

채광이 요구되는 방의 위치를 위치 설정이 가장자리로 제한되지 않고 채광이 가능한 곳의 위치 범위의 한정 제약 조건을 부여함으로써 해결할 수 있다.

4. 구현 및 실험

본 시스템의 구현을 위한 개발 환경은 Windows NT 서버 기반의 펜티엄II PC이다. 서버 측은 CSP 엔진으로 쓰이는 ILOG Solver가 제공하는 C++ 라이브러리를 이용하였으며, 클라이언트는 웹 환경 하에서 브라우저를 이용하기 위한 사용자 인터페이스를 위해서 자바를 사용하였다.

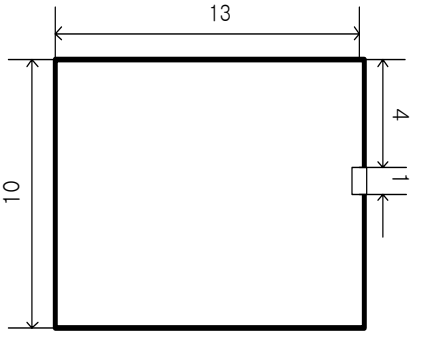


그림 13 주택공간의 설정

표 3 사용자의 초기 요구 조건

자원	최소 길이	최대 길이
현관	2	5
마루	32	38
안방	26	30
침실 1	15	20
침실 2	10	15
화장실 1	8	8
화장실 2	6	6
부엌	15	18

본 시스템의 실험은 [그림 13]과 같은 주택 공간에서 이루어지며, [표 3]과 같은 사용자의 초기 요구 조건을 입력 데이타로 하였다.

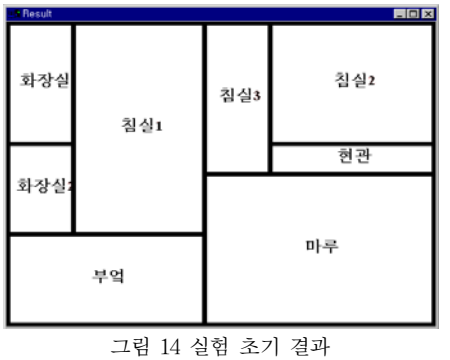
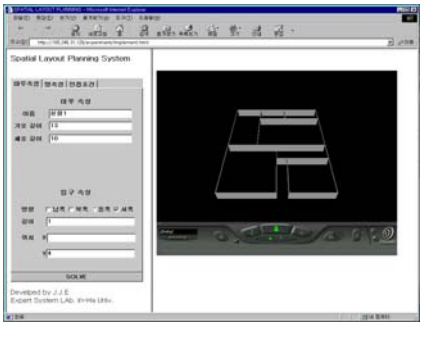


그림 14 실험 초기 결과

[그림 14]는 사용자의 초기 조건에 의해 도출된 2차원 결과이다. 본 시스템의 CSP 엔진에 의해 초기 해를 구할 때까지 약 10 분 정보의 시간이 소요되었다.

(a) 2차원 결과



(b) VRML 결과

그림 15 실험 최종 결과

위의 초기 결과는 침실1에 화장실 2개가 모두 인접해 있으며 침실2에서 침실1로 이동하기 위해 침실3을 거쳐야 한다. 즉, 단지 제약 조건만을 만족시킬 뿐 현실성이 없는 결과이다. 다음 단계로 시스템은 초기 결과보다 사용자 만족도를 높이기 위해서 제약 조건의 수정을 요구하게 된다. 이와 같은 과정을 반복함으로써 사용자 만족도를 최적화하는 결과로 수렴해 가게 되었으며 소요 시간도 줄어들었다. 결국 [그림 15]와 같이 현관을 통해 들어온 다음 마루를 중심으로 모든 방으로 이동할 수 있는 형태로 공간 배치가 이루어졌다. 이 결과는 제약 조건을 만족할 뿐만 아니라 사용자의 만족도도 높으며 현실적인 측면에서도 타당한 결과이다.

5. 결론 및 향후 연구 방향

지금까지 전문가들에 의해 수행되어온 주택 공간 설계를 인공지능 기법을 이용함으로써 자동화하는 시스템이 절실히 요구되어왔다. 본 연구에서는 CSP 기반의 문제 해결 엔진이 장착되었으며 지능형 사용자 인터페이스가 가능한 시스템을 설계하여 구현하였다. CSP 기반의 문제 해결 엔진은 기존의 지식 기반 시스템과 달리 도메인 여과기법을 이용하여 보다 효율적인 성능을 보였다. 또한 지능형 사용자 인터페이스와 같은 사용자와의 효과적인 상호작용 기술을 통해 사용자의 다양한 요구 사항을 수용할 수 있게 하였으며 VRML을 이용하여 웹 환경으로 확장하였다. 그리고 사용자 만족도를 정량적으로 표현하기 힘들며 사용자 만족도 자체가 최적화의 기준이 되기 때문에 시행착오 방법을 이용하여 사용자 만족도를 향상시키도록 하였으며 동적인 제약 조건의 추가, 강화, 삭제, 완화의 과정의 반복을 통해 CSP 엔진의 효율을 향상시켰다. 이러한 시스템을 주택 설계 현장에 도입함으로써 주택 공간 설계 자동화뿐만 아니라 원거리에 있는 사람들이 웹을 통해 자신의 취향에 맞는 가상 주택을 간접적으로 체험할 수 있다.

향후 보완되어야 할 점으로는 건축 설계분야 전문가와의 인터뷰를 통해 건축학적인 지식 베이스를 보완함으로써 좀 더 현실성 있는 결과만을 도출할 수 있게 하고 사용자 의사결정의 효율적 지원에 대한 연구가 추가되어져야 할 것이다. 사용자가 입력하는 인접 조건을 '근방', '비교적'과 같은 표현을 수용할 수 있도록 확장하는 것뿐만 아니라 입력된 사용자 요구 조건들로부터 사용자의 기호(Preference)를 얻어내는 자동화된 사용자 기호 습득(Automated acquisition for user preference)에 대한 연구도 수행할 것이다.

감사의 글

본 연구는 인하대학교 1998년도 연구비 지원에 의하여 수행되었으며 이에 감사를 드립니다.

참 고 문 헌

[1] 전승범, 오윤상, 조근식, "제약만족해결기법을 이용한 공간계획 시스템의 설계 및 구현", 한국정보처리학회 추계 학술발표논문집, 1998

[2] L. B. Kobacs, "Logic Programming systems development for floor plan design by dissectioning Research," Report 87/13, Department of Computer Science, University of Copenhagen, 1987

[3] H. Kazuyoshi, K. Chikahiro, "A Floor Planning System Using Constraints Logic Programming," Technical Report, Tokyo Gas Co., Ltd., No. 93-07, 1993

[4] D. Vasant, R. Nicky, "Integer Programming vs. Expert Systems: An Experimental Comparison," Communication of the ACM, Vol. 33, Number 3, 1990

[5] L. B. Kobacs, "Knowledge based floor plan design by space partitioning : A logic programming approach," Artificial Intelligence in Engineering, 1992

[6] A. K. Mackworth, "Consistency in Networks of Relations," American Association for Artificial Intelligence, 1977

[7] B. A. Nadel, "Constraints Satisfaction Algorithm," Computational Intelligence 5, 1989

[8] D. Frost, R. Dechter, "Insearch of the best constraint satisfaction search," American Association for Artificial Intelligence, 1994

[9] J. E. Beasley, "Algorithms for Unconstrainted Two-Dimensional Guillotine Cutting," Journal of Operational Research Society, 36, 1985a

[10] E. Tsang, *Foundation of Constraint Satisfaction*, pp79-99, Academic Press, 1993

정 재 은

1995년 ~ 1999년 인하대학교 기계공학과/전자계산학과 졸업(복수전공). 1999년 ~ 현재 인하대학교 전자계산학과 석사과정. 관심분야는 지능형 소프트웨어 에이전트, CSP, Robotics, intelligent data analysis

전 승 범

1993년 ~ 1997년 인하대학교 전자계산학과 졸업. 1997년 ~ 1999년 인하대학교 전자계산공학과 대학원(석사). 1999년 ~ 현재 거림시스템(주) 거림연구소 연구원.

조 근 식

1978년 ~ 1982년 인하대학교 전자계산학과 졸업. 1983년 ~ 1985년 Queens College/CUNY 전자계산학 석사. 1985년 ~ 1991년 City University of New York 전자계산학 박사. 1992년 ~ 현재 인하대학교 전자계산공학과 부교수 재직중. 1997년 ~ 현재 한국지능정보시스템학회 논문지 편집위원장. 관심분야는 지능형 소프트웨어 에이전트, 전자상거래, 전문가 시스템, 지식 기반 스케줄링, Constraint Logic Programming언어