論文 한국정밀공학회지 제14권 제9호 (1997년 9월)
Journal of the Korean Society of Precision Engineering Vol. 14. No. 9. September. 1997.

Simulated Annealing을 이용한 제약 네트워크에서의 제약 충족 방식에 관한 연구

차 주 헌*, 이 인 호**, 김 재 정***

Constraint Satisfaction Algorithm in Constraint Network Using Simulated Annealing Method

Joo-Heon Cha*, In-Ho Lee**, Jay J. Kim***

ABSTRACT

We have already presented the constraint satisfaction algorithm which could solve the closed loop problem in constraint network by using local constraint propagation, variable elimination and constraint modularization. With this algorithm, we have implemented a knowledge-based system (intelligent CAD) for supporting machine design interactively. In this paper, we present newer constraint satisfaction algorithm which can solve inequalities or under-constrained problems in constraint network, interactively and efficiently. This algorithm is a hybrid type of using both declarative description (constraint representation) and optimization algorithm (Simulated Annealing), simultaneously. The under-constrained problems are represented by constraint networks and satisfied completely with this algorithm. The usefulness of our algorithm will be illustrated by the application to a gear design

Key Words: Under-constrained(제약부족), Inequality Constraint(부등식 제약), Inference Engine(추론엔진), Intelligent CAD(지능형 CAD), Knowledge-Based System(지식베이스 시스템)

1. 서 론

공학에서의 설계란 "인간의 필요를 충족시키기 위한 선택의 과정"으로 정의된다. 일반적으로 설계는 크게 창의적 설계와 정형적 설계로 나눌 수 있다. 전자인 창의적 설계는 인간 고유의 영역으로 그 어떠한 것으로도 침해할 수 없는 부분이다. 그러나 후자의 정형적 설계에는 지금까지 축적되어온 정형화된 설계자료나 설계경험, 설계대상에 대한 지식 등으로 구성되어 있어, 이를 컴퓨터 내부의 지식 베이스로 구축해두면 컴퓨터 이용에 의한 설계작업의 합리화 및 효율화, 그리고 설계작업의 획기적 단축이 가능해진다[1, 2]. 또한 설계작업의 합리화 및 효율화뿐만 아니라, 인간이 반복적으로 수행해야 하는 단순한 설계작업에서 벗어나 좀더 창의적인 설계에 몰두할 수 있는

* 한국과학기술연구원 CAD/CAM연구센터
** 한양대학교 대학원
*** 한양대학교 기계공학부

환경을 제공해준다.

그러나 최근까지도 컴퓨터를 이용한 설계는 기껏해야 설계대상의 일부분에 대한 단순한 계산과정이나 기하형상에 대한 표현과 그 정의에 불과하였다. 이와 같이 컴퓨터 이용 설계인 CAD의 적용 범위가 단순히 설계대상의 도면이나 기하학 정보의 표현에만 머물러 있어, 최근 이러한 형상정보가 생성되기 이전에 이용되는 설계대상이나 설계과정에 관한 수많은 설계지식을 체계적으로 활용함으로써 설계작업을 효율적으로 지원해주는 인텔리전트 CAD의 구축에 관한 연구가 활발하게 진행되고 있다[3, 4, 8, 9].

인텔리전트 CAD 시스템은 설계과정 전체를 시스템이 자동으로 수행해주는 자동설계와는 그 의미가 다르며, 어디까지나 설계자와 시스템이 서로 협력하여 설계를 진행해가는 대화형의(Interactive) 설계지원 시스템이다. 이와 같은 지능형 시스템 구현을 위해서는 설계지식 표현으로서 규칙 베이스나 객체지향, 제약 프로그래밍 등과 같은 인공지능 및 지식 정보처리 기술의 도입이 필요하다.

본 연구에서는 이러한 설계지식 표현 방법 중 제약표현을 이용하여 인텔리전트 CAD 시스템의 주요 구성모듈의 하나인 강력한 추론엔진 시스템을 구축하는 것을 목표로 하고 있다. 그러나 기존의 제약 네트워크 표현에서는 부등식 제약이나 제약부족(Under-constrained) 문제를 효과적으로 해결할 수 있는 제약충족 방식이 아직 제안되고 있지 못하다. 실제로 대부분의 시스템에서는, 추론한 결과치가 그 조건을 만족하고 있는지를 단순히 체크하는 것으로 부등식 제약을 사용하거나[1, 5], 별도의 최적화 모듈과의 연계에 의해 그 해를 일괄적으로 구하는 방법 등을 사용하고 있다[6, 9].

저자는 이미 제약 표현에 객체지향의 모듈화 개념을 도입함으로써 큰 규모의 복잡한 설계대상에도 적용 가능한, 그리고 동시에 제약 네트워크의 폐루프 문제를 효과적으로 해결할 수 있는 새로운 제약충족 알고리즘을 제안한 바 있다[1].

본 연구에서는 이미 제안한 제약충족 알고리즘에 인공지능의 최적화 기법인 SA(Simulated Annealing)법을 도입하여, 전술한 부등식 제약과 제약부족 문제를 효과적으로 해결할 수 있는 더욱 발전된 제약충족 알고리즘을 제안한다. 즉 제약 표현의 선언적(Declarative) 기술과 최적화 기법인 SA법을 동질의(Homogeneous) 제약 네트워크 표현으로 결합시켜 서로의 강점과 약점을 이용·보완시킨 Hybrid형 알고리즘이다. 여기에서는 이를 기반으로 실제로 인텔리전트 CAD 시스템의 추론 시스템을 구현해보고, 치차 설계에 적용해봄으로써 본 알고리즘 및 개발 시스템의 유효성을 확인해본다.

2. 제약 네트워크

2.1 제약 네트워크에 의한 설계대상의 표현

제약 표현에서는 설계대상의 설계변수 노드와 설계제약 노드로 구성되는 네트워크 형태로 표현된다. Fig.1은 제약 네트워크의 예를 나타낸다. 그림에서 사각형 노드는 설계제약 노드를 나타내며, 원형노드는 설계제약에 포함되는 설계변수 노드를 나타낸다. 각 노드간의 관계는 쌍방향의 성질을 가지며, 설계변수 노드는 동일한 레벨로 표현되기 때문에 설계과정의 순서에 관계없이 어느 설계변수의 노드로도 입출력이 가능한 선언적 표현의 특징을 지닌다. 선언적 기술은 설계지식이 제어 프로그램과 분리되어 처리되기 때문에 설계지식의 관리가 용이하고 시스템 구현이 용이한 장점을 가지고 있다.

예를 들어 Fig.1(a)의 제약 노드 [A+B+C+D=10]

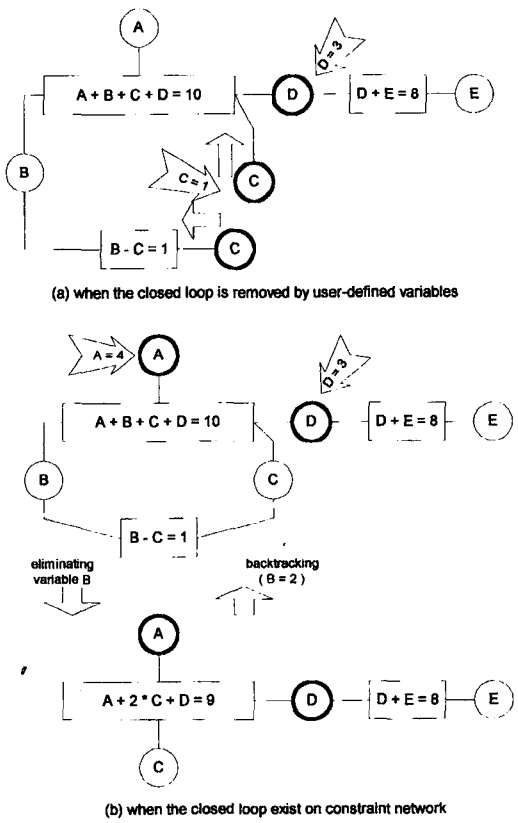

Fig. 1 Constraint satisfaction in constraint network

을 순서적(Procedural) 표현으로 기술하면, 설계변수의 개수만큼의 함수를 필요로 한다. 즉 A = f1 (B, C, D), B = f2 (C ,D ,A), C = f3 (D, A, B), D = f4 (A, B, C)와 같은 4개의 함수가 필요한 것이다. 여기에서 fn은 함수를 나타내며, 괄호 안의 알파벳은 입력변수를 나타낸다.

반면에 제약 네트워크의 선언적 표현에는 이러한 부가적인 함수가 필요 없어 설계대상의 표현과 시스템의 구현이 간단 명료해지는 것은 물론, 크고 복잡한 설계대상의 표현도 가능한 장점을 가진다.

2.2 제약전파와 변수소거에 의한 제약충족

제약 네트워크상에 폐루프가 존재하지 않은 경우에는 제약전파에 의해 그 해를 손쉽게 구할 수 있지만, 폐루프가 네트워크상에 존재하는 경우에는 그 해를 정확하게 구하기 곤란하다[7]. 따라서 본 연구에서는 제약 네트워크상에 폐루프가 존재하는 경우 사용자 정의 노드의 분리 및 변수소거에 의해 제약충족을 수행한다.

폐루프를 포함하는 제약 네트워크에서의 제약충족 방식은 다음과 같이 두 가지 경우를 생각할 수 있다. 첫 번째는, Fig.1(a)에서와 같이 사용자 정의 속성 노드 C가 폐루프상에 존재하는 경우를 생각할 수 있다. 이때 사용자 정의 변수 노드 C는 설계 변수치가 이미 설정되어 있는 상수 노드이므로, 이를 두개의 상수 노드로 분리하면 네트워크상의 폐루프는 자연스럽게 제거되어, 제약전파에 의해 미지수 노드 B, D, A의 값이 차례로 결정된다.

두 번째는 Fig.1(b)에서와 같이 폐루프가 제거되지 않고 그대로 네트워크 상에 존재하는 경우이다. 이 경우에는 아래 그림과 같이 수식처리에 의해 변수 B를 소거함으로써 네트워크상의 폐루프를 제거한다. 이때 변수소거 과정의 이력은 별도로 기억되며 나중에 소거된 변수를 회복시키기 위한 백트래킹에 이용된다. 이와 같이 변수소거에 의해 폐루프가 제거되어 제약전파에 의해 설계변수 노드 D와 C의 값이 차례로 결정되며, 소거된 변수 이력의 백트래킹에 의해 최종적으로 미지수 노드 B의 값도 자동으로 결정된다.

여기에서 채택한 제약충족 방식은 국소적인 제약전파와 변수소거 방식을 하나의 네트워크 표현에 의해 결합시킴으로써 폐루프 문제와 비선형 문제를 포함하는 설계제약을 대화형으로 효과적으로 해결할 수 있는 장점을 가지고 있다. 반면에 이러한 제약전파와 변수소거의 결합 방식으로는 부등식 제약과 조건이 부족한 제약부족 문제를 효과적으로 해결할 수 없는 단점도 지니고 있다.

본 연구에서는 이와 같은 부등식 및 제약부족 문제를 해결하기 위해 제약 표현의 선언적 표현에 인공지능(Artificial Intelligence, AI) 기술의 최적화 기법인 SA법을 결합시킨 새로운 제약충족 방식을 제안한다.

3. SA법에 의한 제약 충족

3.1 SA법의 채택 배경

일반적으로 기존의 최적화 기법은 빠른 시간내에 최적화가 이루어지지만 탐색한 최적점이 국부해일 가능성이 있기 때문에 그 입력 데이터를 바꾸어 가면서 반복하여 재실행시켜야 하는 문제점이 있다. 반면에 AI 최적화 기법은 다소 최적화 시간은 길지만, 초기치 영향이 비교적 적어 재실행의 불편함이 없고, Random Process의 채용으로 국부해에서의 탈출이 가능하기 때문에 최적해 탐색이 기존의 최적화 기법에 비해 상대적으로 우수한 장점을 지닌다.

본래 인텔리전트 CAD 시스템은 사용자와 시스템이 상호간의 대화를 통하여 설계를 진행하는 방식이므로, 최적해를 얻기 위해 여러번 반복하여 재실행해야 하는 기존의 최적화 기법은 채용이 곤란하다. 따라서 본 연구에서는 초기치에 영향을 비교적 적게 받으면서 최적해 탐색이 우수한 SA법을 채용하였다.

한편 AI 최적화 기법의 하나인 유전자 알고리즘(Genetic Algorithm, GA)은 설계변수들을 2진 코드(Binary Code)로 변환해야 하는 번거러움이 있어 대화형 시스템인 제약 네트워크 표현에 그대로 적용하기 곤란하고, SA법에 비해 국부해에 빠질 가능성이 비교적 높은 편이어서 본 방식에는 적용할 수 없었다.

3.2 SA법의 이론적 배경[10-12]

SA 알고리즘은 느린 냉각에 의한 금속의 어닐링 과정이나 용액의 결정화 과정에서 유추된 Hill Climbing법의 변형된 알고리즘이다. 즉 고온에서 금속 분자는 충분한 운동 에너지를 가지고 각 분자에 대해 서로 자유롭게 움직이다가 천천히 냉각되면서, 그 에너지를 잃게 되어 완전히 배열된 결정이 만들어진다.

SA 알고리즘에서 변수의 값은 분자를 나타내며, 비용함수(Cost Function)는 에너지를 나타낸다. SA법은 자연계의 느린 냉각 과정에 대한 모델이기 때문에, 최적해를 얻기 위해서는 시스템이 결빙(Freeze)될 때까지 그

냉각일정을 조정하는 것이 중요하다.

자연계의 냉각 과정에서는 낮은 에너지 상태에서 높은 에너지 상태로의 이동이 가능하다. 이와 같은 현상에 의해 국부해(Local Minimum)로부터의 탈출이 가능하기 때문에 기존의 최적화 기법에 비해 최적해(Global Minimum)에 도달할 수 있는 높은 확률을 가진다. SA법은 이러한 과정을 Metropolis 테스트[10, 11]에 의해 모델화함으로써 그 최적해를 얻을 수 있는 확률적 보장을 가진다. 이때 Metropolis 테스트는 Boltzmann 분포의 확률을 따른다.

3.3 SA법의 문제 해결 과정

SA법은 여러 종류의 문제에 적용이 용이한 범용 알고리즘이다. 따라서 특정한 문제에 적용하려면 문제에 적합한 목적함수와 변수를 설정해야 한다. 변수와 목적함수가 문제설정에 맞지 않으면 계산이 비효율적이 됨은 물론 정확한 해를 구하지 못하게 되는 경우가 많다.

현재의 변수치와 비교하기 위한 새로운 변수치는 기본적으로 이전의 변수치에 이웃하면서 제약조건을 만족하는 값을 무작위로 선택한 값이다. 선택된 새로운 변수를 목적함수에 대입하여 그 비용을 구한다. 이 새로운 변수치가 비용을 줄이는 방향으로 변경되었거나 Metropolis 테스트를 통과하게 되면 승인되어 현재의 변수치로 결정된다. 시스템이 열적 평형에 도달하게 되면 현재의 변수치를 최적해로 결정하여 종료하고, 아직 평형상태가 아니면 다시 새로운 변수치를 찾아 테스트를 반복한다. 이해를 돕기 위한 SA법의 흐름도를 Fig. 2에 나타내었다.

SA법에서는 냉각일정 조정과 초기온도 설정이 최적해의 신뢰성에 많은 영향을 끼친다. 최적해를 찾기 위해서는 충분히 느린 일정을 가져야 하지만, 지나치게 느린 냉각일정은 해를 찾는 시간을 지나치게 요구한다. 또한 초기온도를 너무 낮게 설정하면 해는 최적해를 찾지 못한 채 국부해에서 냉각된다. 따라서 본 시스템에서는 범용성을 높이고 사용자 조정을 줄이기 위하여 시스템 스스로 적당한 초기온도를 설정하도록 하였으며, 초기냉각 속도는 비교적 빠르고 점차 느린 냉각속도를 가지게 하는 방식을 채택했다. 즉, 적합한 초기온도를 설정하기 위하여 시스템은 임의 온도에서 시작하여 Metropolis 테스트를 통과하는 비율이 일정 비율을 넘을 때까지 일정 간격으로 온도를 높이게 된다. 승인의 수가 일정 회수를 넘으면 이 온도를 초기온도로 설정한다. 그리고 냉각 일정은 이전 온도에 일정한 인자를 곱하여 새로운 온도로 설정하는 일정을 가지도록 하였다.

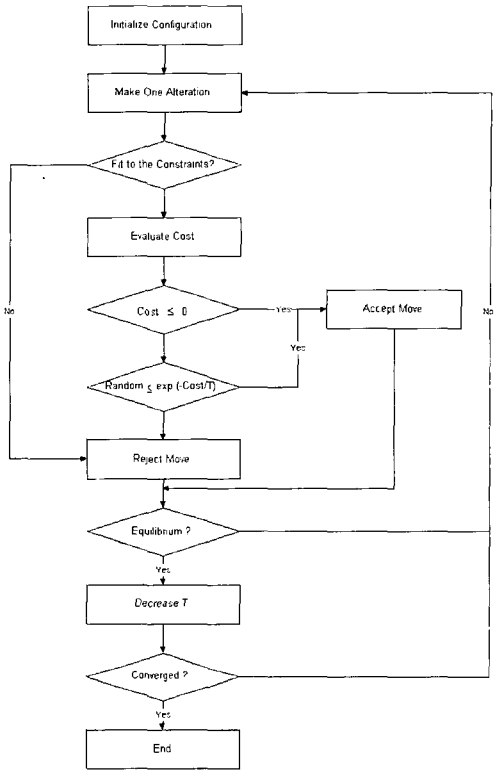

Fig. 2 Flow chart of simulated annealing algorithm

4. Hybrid형 제약충족 방식

4.1. 제약 네트워크와 SA의 결합

앞에서 기술한 제약 네크워크 표현과 SA법은 각각 여러 가지의 우수한 장점을 가지고 있기 때문에 두 알고리즘을 결합시켜 상호 보완적인 새로운 알고리즘을 만들 수 있다.

제약 표현은 앞서 언급한 바와 같이 설계 대상을 제약 네트워크로 표현함으로써 설계제약의 선언적 표현이 가능하다. 이 선언적 표현에 의해 설계제약을 변수의 개수만큼 부가적인 함수를 필요로 하지 않아 설계 대상을 표현할 때 시스템의 구현이 용이하며 그 유연성도 크다. 또한 선언적 표현에 의해 추론 및 계산 과정이 일괄적으로 수행되지 않기 때문에 설계자와 시스템과의 상호 대화가 가능한 대화형 제약충족 방식을 구현할 수 있다. 더구나 설계 과정의 경험적 지식을 제약 네트워크 표현에 삽입함으로써 기능의 확장이 가능하며, 설계제약의 해를 찾기

위해 반복하여 계산하는 기법이 아니기 때문에 최적화 기법에 비해 매우 빠르고 정확하게 해를 구할 수 있다.

그러나 제약 네트워크에 의한 표현은 부등식 제약이나 제약부족 문제의 경우 만족할만한 변수의 값을 제시하지 못하기 때문에, 설계자와의 대화를 통하여 이 문제를 해결하였으나 설계제약이 많은 복잡한 설계 대상인 경우 이를 일일이 설계자가 확인해주어야 하는 문제점이 있다.

SA법은 여러 종류의 문제에 쉽게 적용할 수 있고 초기치가 최적해에 비교적 크게 영향을 끼치지 않아서 범용의 알고리즘으로 쓰일 수 있다. 또한 제약의 적용이 쉽고 최적해에 도달하는 우수한 성능이 여러 분야에서 증명된 바 있다. 그러나 SA법은 냉각일정이 충분히 느리도록 계산을 반복하는 방식이므로 변수가 많아질수록 해를 구하는 시간이 급증하고, 구한 해가 최적해가 아닐 가능성을 배제할 수 없다. 또한 설계 대상을 선언적으로 표현할 수가 없는 단점을 지닌다.

따라서 본 연구에서는 이 두 알고리즘을 결합하여 상호 보완적으로 제약충족을 수행할 수 있는 Hybrid형 알고리즘을 제안하고, 이를 기반으로 실제로 제약충족 시스템을 구현하여 그 유효성을 검증해본다.

4.2 Hybrid형 제약충족 방식

설계 제약식이 입력되면 제약 네트워크가 생성되어 초기화면이 사용자에게 제시된다. 사용자의 필요에 따라 사용자 정의 변수치, 즉 설계 사양을 입력하면 시스템은 우선 2.2절의 제약전파와 변수소거에 의한 제약충족 방식에 의해 각 설계변수의 해를 구하게 된다. 이와 같은 과정에 의해 복잡한 설계제약은 간단한 설계제약으로 압축되게 되며, SA법에 의해 탐색해야 할 탐색 영역이 전체적으로 줄어들게 되어 최적해 탐색 시간이 크게 단축되어 진다.

이때 미결정 설계변수가 발견되어 시스템은 자동적으로 압축되어진 설계 제약식을 나머지 변수들에 적용하여 SA법에 의해 미결정 설계변수의 해를 구하게 되며, 최종적으로 폐루프 제거를 위해 소거되었던 설계변수들이 있는 경우에는 그 소거 이력에 의해 제약 네트워크상에 회복되어 제약전파에 의해 그 값이 결정되어 진다. 이렇게 하여 하나의 제약모듈에서의 모든 설계변수는 모듈 내의 설계 제약을 만족하도록 그 값들이 결정되어 진다. 결정된 설계 변수치는 제약모듈간의 공통 변수를 통하여 다른 제약모듈로 제약전파되어 앞의 제약충족 방식을 반복하게 된다. Fig. 3은 Hybrid형 제약충족 시스템의 전체 흐름도를 나타낸다.

4.3 Hybrid형 제약충족 방식의 적용 예

본 연구에서의 제약은 등식 제약, 부등식 제약, 그리고 각 설계변수의 범위 등 3가지 종류로 구분된다. 각 변수의 범위도 부등식 제약의 범주에 속하나 제약 네트워크로 표현되지 않고 구한 변수치의 적합성 판정에 사용된다.

Fig. 4는 부등식 제약을 포함하는 제약 네트워크의 예를 나타낸다. 시스템은 먼저 제약전파와 변수 소거 등의 방식을 통하여 제약충족을 수행한다. Fig. 4는 사용자 정의 변수 중 하나가 폐루프를 구성하고 있는 경우이다. 먼저 사용자 정의 노드 C는 상수 노드이므로 두개의 노드로 분리되어 네트워크의 폐루프가 사라지게 된다(①). 분리된 노드 C의 값이 제약 노드 "B - C = 1"에 전파됨으로써 변수 노드 B의 값이 결정된다(②). 동시에 사용자 정의 노드 D에 의해 미지수 노드 F의 값이 결정된다(③\).

다음에 제약 네트워크 상에서 아직 결정되지 않은 변수 A, E와 부등식 제약 조건 "A < F - 2"와 "E < D + 1"을 검색하여 SA법에 의해 나머지 변수의 최적해를 구하게 된다. 보다 효과적인 SA법의 사용을 위해, 사용자는 시스템과의 대화를 통하여 해를 구하지 못한 변수에 대한

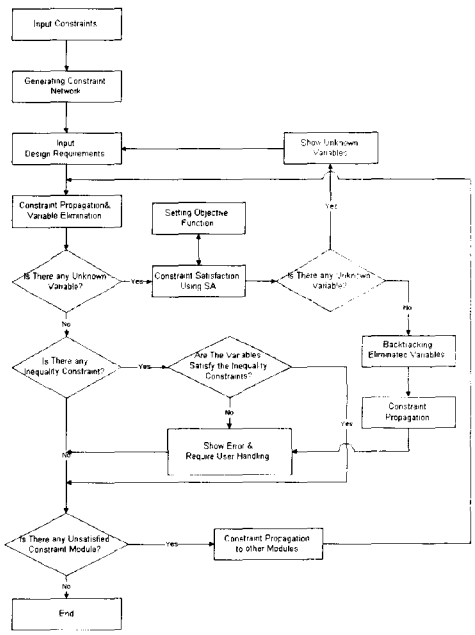

Fig. 3 Flow chart of hybrid constraint satisfaction algorithm

목적함수를 자유롭게 설정할 수 있다. Fig. 4의 적용 예에서는 목적함수를 $f = A^2 + E^2$로 설정하여 최소화하도록 하였으며(④), SA법에 의해 제약충족을 수행하여 다음과 같이 변수 A와 E에 대한 최적치를 얻을 수 있다.

$A = 0.9979248047, \quad E = 1.00207517950$

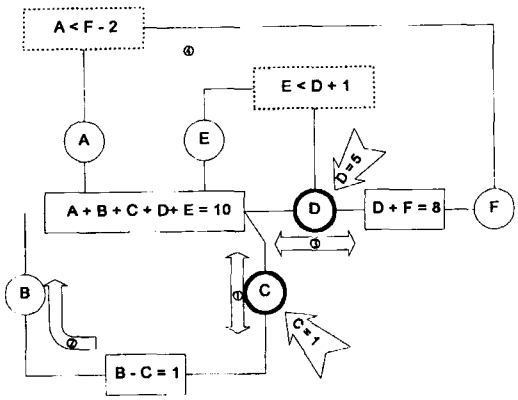

Fig. 4 Example of constraint satisfaction algorithm

5. Hybrid형 제약충족 시스템 구현 및 적용 사례

본 연구에서의 Hybrid형 제약충족 시스템은 IBM PC 586의 Windows NT에서 객체지향 언어인 Visual C++와 MFC에 의해 구현되었다.

여기에서는 본 시스템의 유효성을 검토하기 위해 실제의 설계 예로서 굽힘응력을 고려한 치차 설계에 적용해보았다. 치차는 두개의 제약모듈[1] 즉 기하학 제약모듈과 굽힘응력 제약모듈로 구성되며 그 설계제약은 다음과 같다.

- 기하학 제약모듈에서의 설계제약 :

$a = (d_1 + d_2)/2 \quad d_2 = z_2 * m$

$u = z_2 / z_1 \qquad d_1 = z_1 * m$

- 굽힘응력 제약모듈에서의 설계제약 :

$\sigma_1 \geq F_1 * Y_1 / b / m \qquad \sigma_2 \geq F_2 * Y_2 / b / m$

$u = n_1 / n_2$

$F_1 = 1.91 * 10^7 * P / d_1 / n_1$

$F_2 = 1.91 * 10^7 * P / d_2 / n_2$

단, 여기에서 설계변수인 a : 중심거리, d : 피치직경, u : 감속비, m : 치차모듈, z : 잇수, P : 전달동력, σ : 허용응력, n : 회전수, Y : Lewis 계수, b : 치폭, F : 접선력을 각각 나타낸다.

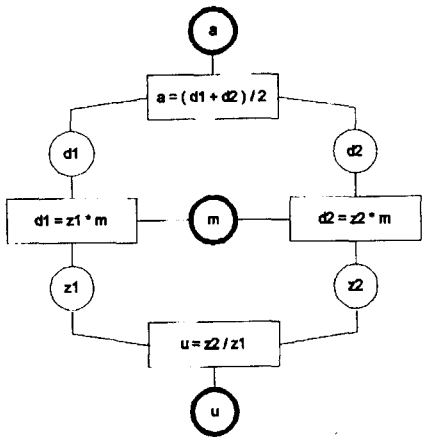

Fig. 5 Constraint network representation of geometry constraint module

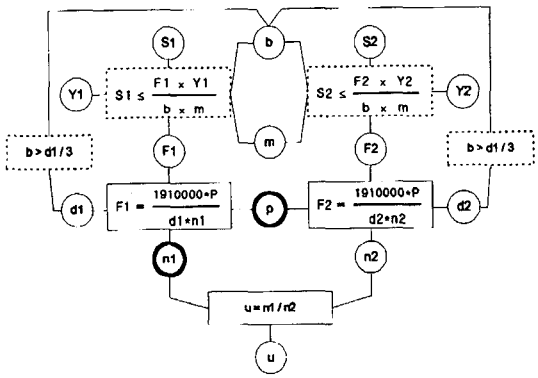

Fig. 6 Constraint network representation of bending stress constraint module

이러한 설계제약이 시스템에 입력되면 시스템 내부에서 설계 대상의 제약 네트워크를 생성하여 관리한다. Fig. 5와 6은 각각 기하학 제약모듈과 굽힘 응력 제약모듈을 제약 네트워크로 표현한 것이다.

시스템은 생성된 제약 네트워크를 바탕으로 Fig. 7와 같은 초기상태의 화면을 제약모듈별로 생성하여 사용자에게 제시해주며, 이 초기화면을 이용하여 사용자는 시스템과 대화를 하면서 설계 작업을 수행한다. 입력될 설계사양은 다음과 같다.

중심거리 a = 150mm　　치차 모듈 m = 5mm

변속비 u = 3　　회전수 n_1 = 1500rpm

전달 동력 P = 21kW

그리고 설계변수의 범위는 다음과 같이 설정한다.

$d_1/3 < b < d_2/5, 20 \leq \sigma_1 \leq 100, 20 \leq \sigma_2 \leq 100$

위의 설계 사양을 초기화면에 입력하여 본 연구에서 제안한 제약충족 방식에 의해 모든 제약을 만족하도록 모든 설계 변수치를 구하게 된다.

한편 제약부족 조건은 "설계변수의 개수 〉 제약식의 개수 + 입력변수(설계사양)의 개수"로 정의된다. 따라서 본 연구에서 적용한 사례는 설계변수가 15개, 제약식이 7개, 입력변수가 5개이기 때문에 제약부족 조건임을 알 수가 있다.

우선 제약전파 및 변수소거 방식에 의해 기하학 제약모듈에서의 제약충족을 행하게 된다. 이 제약충족 결과에서 결정된 설계 변수치는 다음과 같다.

피치 직경 : $d_1 = 75.00, d_2 = 225.00$

잇수 : $z_1 = 15, z_2 = 45$

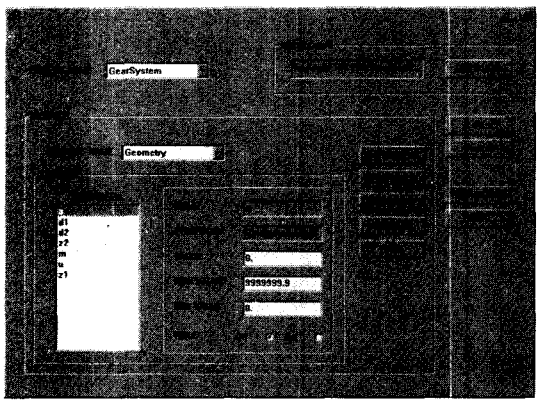

Fig. 7 Initial state window

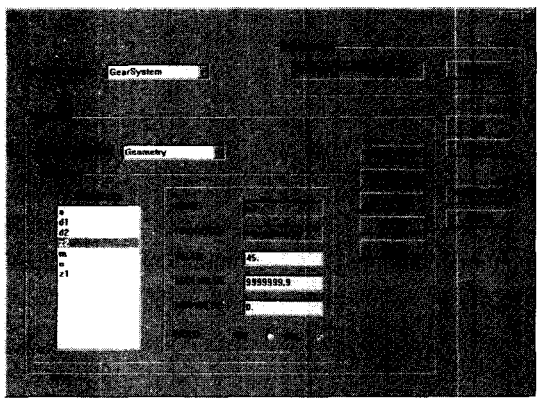

Fig. 8 Result of constraint satisfaction in geometry constraint module

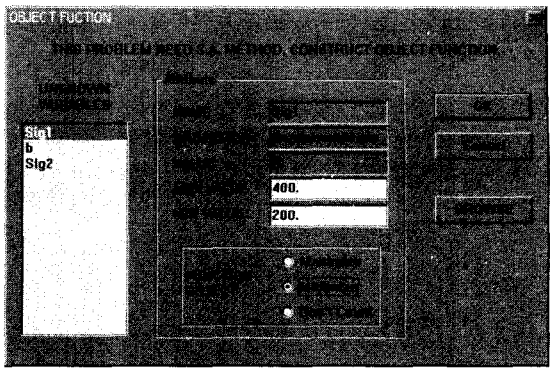

Fig. 9 Editor for objective functions

Fig. 10은 기하학 제약모듈에서의 제약충족 결과를 나타낸다. 여기에서 잇수 z_2가 45로 결정되었음을 알 수 있다.

이렇게 기하학 모듈에서 결정된 설계 변수치는 두 제약모듈간의 공통 설계변수인 m, d_1, d_2, u를 통하여 굽힘응력 모듈로 제약전파된다. 이때 전파된 설계변수 노드는 상수 노드로서 폐루프 제거에 의해 제약충족을 신속하게 수행해준다.

이때 제약전파와 변수소거에 의해 충족되지 않은 변수와 부등식 제약은 SA법에 의해 제약충족이 수행된다. 효과적인 SA법 제약충족이 수행되기 위해서는 적절한 목적함수의 설정이 매우 중요하다.

이와 관련하여 본 연구에서는 다음의 두 가지 방식을 채택한다. 먼저 Fig. 9과 같이 목적함수 편집 화면을 사용자에게 제시하여 사용자가 미결정 변수와 관련한 목적함수를 직접 대화를 통하여 설정할 수 있는 방식이다. 한편 기계설계의 경우 설계 대상에 따라 목적함수가 다를 수 있다. 예를 들면 설계 대상의 중량을 줄이는 혹은 제작비용을 줄이는 방향으로 목적함수를 설정할 수도 있어, 설계자의 의도나 경험이 중요한 요소가 된다. 따라서 이러한 경우에는 본 연구에서는 목적함수를 하나의 설계 지식으로서 설계제약과 함께 기술하는 방식을 채택한다. 본 사례연구에서는 목적함수를 $\mathbf{MIN.\ F = b^2 - S1^2 - S2^2}$ 으로 설정하였다.

목적함수가 설정되면 시스템은 나머지 미결정 설계 변수치를 SA법에 의해 결정하여 그 결과를 화면에 출력한다. 이때 결정된 설계 변수치는 $b = 25.03mm$, $\sigma_1 = 83.34MPa$, $\sigma_2 = 63.50MPa$이다. Fig. 8은 굽힘응력 제약모듈에서의 제약충족 결과를 나타내며, 현재 치폭 b = 25.03mm를 출력하고 있다.

이와 같이 본 연구에서 개발한 제약충족 시스템은, 사용자가 시스템과의 대화를 통하여 부등식 및 제약부족 조건을 포함한 모든 설계제약을 만족하도록 설계 대상의 모든 설계변수치를 효율적으로 결정할 수 있다.

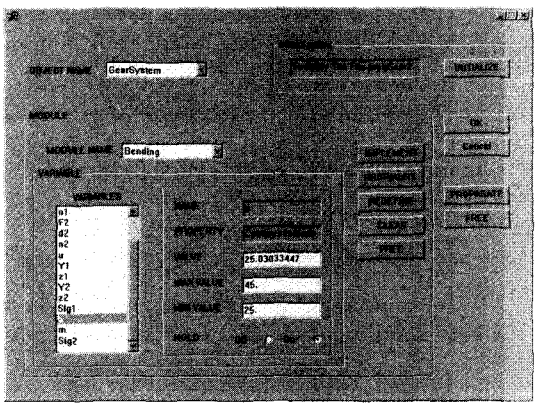

Fig. 10 Result of constraint satisfaction in bending moment module

6. 결 론

본 연구에서는 설계지식의 표현 방법 중 하나인 제약표현을 이용하여 인텔리전트 CAD 시스템의 구성요소의 하나인 강력한 추론엔진을 구축하는 것을 목표로 하여, 제약 네트워크에서의 제약전파와 변수소거에 의한 제약충족 알고리즘에 인공지능의 최적화 기법인 SA법을 도입하여 부등식 제약과 제약부족 문제를 효과적으로 해결하는 새로운 형태의 제약충족 알고리즘을 제안하였다. 이는 제약 표현의 선언적 기술에 최적화 기법인 SA법을 결합시켜 서로의 강점과 약점을 이용·보완한 Hybrid형 방식이다.

이를 기반으로 기계설계를 효율적으로 지원할 수 있는 제약충족 시스템을 실제로 구현해보았으며, 부등식 제약과 제약부족 조건을 가진 치차 설계에 실제로 적용해봄으로써 본 연구에서 제안한 제약충족 알고리즘과 구현 시스템의 유효성을 확인하였다.

참 고 문 헌

1. 차주헌, "Design Constraints를 이용한 기계설계 지원 시스템", '96한국전문가시스템학회 추계학술대회 논문집, pp.103, 1996.
2. 차주헌, "형상 패턴 인식을 이용한 설계자료의 자동 탐색", 대한기계학회 논문집(A편), 제21권, 제4호, pp.634, 1997.
3. Cha, J.H., Yokoyama, M. and Itaru, O., "A Knowledge-Based System for Mechanical CAD", ICED'93, pp.1335, 1993.
4. Cha, J.H. and Yokoyama, M., "A Knowledge-Based System for Mechanical CAD", ICED'95, pp.1382, 1995.
5. Yokoyama, T. and Sazuka, H. "A Constraint-Based and Object-Oriented Knowledge Representation System", 日本 情報處理學會 論文集, Vol.31, No.1, pp.68, 1990.
6. Fujita, K., Akagi, S., Hase, H., Nakatogawa, T. and Takeuchi, M., "Hybrid-Type Approach for Plant Layout Design with a Constraint-Directed Search and a Mathematical OptimizationTechnique", JSME(C), Vol. 58, No.547, pp.967, 1993.
7. Fromont, B. and Sriram, D., "Constraint Satisfaction as a Planning Process", in Gero, J.S.(ed.), Artificial Intelligence in Design'92, Kluwer Acade. Pub., 1992.
8. Young, R.E., Greef, A. and O'Grady, P., "SPARK - An Artificial Intelligence Constraint Network System for Concurrent Engineering", in Gero, J.S.(ed.), Artificial Intelligence in Design'91, 1991.
9. Serrano, D. and Gossard, D.C., "Tools and Techniques for Conceptual Design", in Tong, C. and Sriram, D. (ed.), Artificial Intelligence in Engineering Design, Academic Press Inc., 1992.
10. Press, W.H., Elannery, B.P., Teukolsky, S.A. and Vetterling, W. T., "Numerical Recipes in C", Cambridge Univ. Press, 1988.
11. P.J.M. van Laarhoven, E.H.L. Aarts "Simulated annealing: Theory and Applications", D. Reidel Publishing Co., 1987.
12. Wong, D.F., Leong, H.W. and Liu, C.L. "Simulated annealing for VLSI design", Kluwer Academic Publishers, 1998.