韓國컴퓨터情報學會 論文誌
第14卷 第4號, 2009. 4.

2009-14-4-1-1

# 동적계획법을 이용한 철근가공용 소프트웨어의 구현

김 성 훈*

# An Implementation of Cutting-Ironbar Manufacturing Software using Dynamic Programming

Seong hoon Kim*

## 요 약

이 논문에서는 철근 절단 작업의 계획 문제를 동적 계획법으로 해결하여 근사 최적의 절단 계획을 생성하도록 하는 소프트웨어의 구현을 다룬다. 일반적으로 실제 절단 작업에 요구되는 제약사항을 반영하여 최적의 자재 절단 문제의 해를 얻는 알고리듬의 설계가 필요하다. 하지만, 이것은 다중 규격의 1차원 자재 절단 문제를 풀어야 하는 것으로, 최적의 해를 얻는 선형계획법은 폭발적인 계산량과 기억용량의 한계로 적용하기 어렵다. 이러한 한계를 해결하기 위하여, 동적계획법에 근거하여 자재 절단 문제를 재구성하고, 휴리스틱을 적용하여 유한 범위의 조합 열에서도 근사 최적의 해를 찾을 수 있는 탐색 기법을 사용한 자재 절단 계획 알고리듬을 제시하였다. 그리고, 자동화된 철근 가공 산업용 소프트웨어는 작업 환경에 맞게 사용이 편리한 그래픽 화면과 사용자 인터페이스가 요구되는데, 공개 소프트웨어를 활용한 GUI 라이브러리 툴킷인 GTK+를 활용하여 이를 구현하였다. 개발된 소프트웨어는 철근 가공의 현장 지식을 바탕으로 휴리스틱 지식을 획득하여 동적계획법에 적용시킨 것으로, 지역 전통 산업과 첨단 IT 산업이 접목된 융합 IT를 시도한 사례 연구이다.

## Abstract

In this paper, we deal an implementation of the software that produces sub-optimal solution of cutting-ironbar planning problem using dynamic programming. Generally, it is required to design an optimization alogrithm to accept the practical requirements of cutting ironbar manufacturing. But, this problem is a multiple-sized 1-dimensional cutting stock problem and Linear Programming approaches to get the optimal solution is difficult to be applied due to the problem of explosive computation and memory limitation. In order to overcome this problem, we reform the problem for applying Dynamic Programming and propose a cutting-ironbar planning algorithm searching the sub-optimal solution in the space of fixed amount of combinated columns by using heuristics. Then, we design a graphic user interfaces and screen displays to be operated conveniently in the industry workplace and implement the software using open-source GUI library toolkit, GTK+.

• 제1저자 : 김성훈
• 투고일 : 2009. 02. 17, 심사일 : 2009. 03. 13, 게재확정일 : 2009. 04. 20.
* 경북대학교 소프트웨어공학과
※ 이 논문은 2008년 한국컴퓨터정보학회 제38차 하계학술대회에서 발표한 논문("동적계획법을 이용한 철근가공 산업용 소프트웨어 개발")을 확장한 것임

▸ Keyword : 자재철단문제(cutting stock problem), 동적계획법(dynamic programming), 철근절단계획(cutting ironbar plan)

# Ⅰ. 서 론

철근 가공 작업은 건설 및 토목 현장에서 사용될 철근을 설계도면을 통하여 필요로 한 철근의 형상과 길이 및 갯수를 산출하고서, 이러한 철근 수요의 요구 사항에 따라 적절한 길이로 철근을 절단하고, 이를 절곡하여 원하는 형상으로 가공하여 공사현장에 납품한다. 현재 많은 가공 산업 현장에서는 이러한 가공 작업 과정에서의 철근 절단 계획을 수립하는 중간 과정을 스프레드쉬트와 같은 단순 태이블 처리 프로그램에 의존하여 수작업으로 하고 있어서 담당 엔지니어의 애로와 함께 비효율이 발생하고 있다. 특히, 납기마감 기한이 짧은 긴급한 주문에 대해 최적의 작업수립을 하지 못하여 많은 고철 발생으로 인한 철근 자원의 낭비를 초래하고 있다.

이러한 문제는 철근 가공 산업뿐만 아니라 섬유 산업이나 제지산업, 유리산업, 가구산업, 등의 산업현장에서 많이 필요로 하는 최적화의 문제로 전형적인 자재 절단 문제(cutting stock problem)라고 알려져 있다[1][2].

이러한 최적화 문제를 푸는 데 많이 사용되는 선형계획법에 의한 해결방법은 최적의 해를 얻을 수 있지만, 실제적인 문제를 적용하는 데 있어 산업현장에서 요구하는 복합적인 제약과 조건을 그대로 적용이 어려운 단점을 안고 있다. 그리고, 이를 그대로 선형계획법에 의해 해를 얻는 경우에 방대한 양의 조합으로 후보열이 생성되어, 메모리 부족 및 계산시간의 한계에 부딪히게 된다. 이를 해결하기 위한 방안으로 최적의 해 대신에 근사 최적의 해를 찾는 방법이 실제로 많이 사용되고 있다.[3][4][5]

철근 절단 계획 수립의 문제는 1차원 자재 형태의 철근에 대한 절단 계획을 세우는 고전적인 자재 절단 문제이지만, 철근 가공 업체로부터 요구된 추가적인 조건이 철근 자재(stock)의 규격이 하나로 동일하지 않고, 여러 단위 길이로 제공되는 다중 규격 자재를 대상으로 하고 있고, 또한 규격이 정해져 있지 않은 난척 자재도 포함되어 있으며, 공급되는 자재 재고량도 종류에 따라서 유한 갯수로 제한되어 있으며, 얻고자 하는 가공 산출물의 종류가 많을 경우에는 20가지 이상의 조합된 결과를 요구하고 있다. 이러한 제약 조건은 Gilmore와 Gomory가 제시한 선형계획법의 해법을 그대로 적용하기에는 메모리량과 계산 시간의 측면에서 한계를 가진다.

이 논문에서는 효율적으로 다중 규격의 자재 절단 문제를 풀기 위하여 동적계획법(dynamic programming)에 근거하여 자재 절단 문제를 재구성하고, 주어진 실제 조건의 문제영역에서 얻을 수 있는 활용 가능한 휴리스틱을 적용하여 유한 범위의 조합 열에서도 근사 최적의 해를 찾을 수 있는 탐색 기법을 사용하는 자재 절단 계획 알고리듬을 사용하였다.[6]

이와 더불어 철근 절단 계획을 자동으로 생성하는 소프트웨어를 GUI방식으로 구현하는데 있어서 개발비용의 절감을 위해 오픈소스 소프트웨어의 개발환경과 도구를 활용할 것을 모색하였다. 이 소프트웨어의 GUI를 설계하는데 있어서 고려되어야 할 사항으로, 철근 절단 생산 현장에서의 사용자와 환경의 제약과 기존 업무와의 연결성을 유지해야 하는 기능적인 요구사항을 만족시키면서, 사용이 편리한 GUI 인터페이스를 설계해야 한다. 이를 위해서 태스크 기반의 GUI 모델링 방법에 의해 화면 구성과 인터페이스 설계를 행하였다. 설계된 GUI는 GTK+ 라이브러리 툴킷으로 빠른 프로토타입을 구현하였다.

본 논문의 구성은 다음과 같다. 2장에서는 설계하려는 소프트웨어의 문제 분석과 관련연구를 기술하였고, 3장에서는 알고리듬을 설계하였으며, 4장에서는 시스템 설계와 개발 환경을 기술하였다. 5장에서 프로토타입의 구현 결과를 고찰하고, 6장에서 결론과 향후의 연구 방향에 대해 기술한다.

# Ⅱ. 문제 분석 및 관련 연구

## 1. 절단 계획 수립 문제의 분석

철근 절단 계획 수립의 문제는 〈그림 1〉에서 보는 것처럼, 주문 사항에 따라 재고 철근의 투입량을 최소로 하면서 주문 철근을 생산하기 위해, 가공 작업에서 발생되는 고철의 양을 최소화하는 절단 계획을 생성하는 것이다. 이때, 공급되는 재고 철근은 길이가 규격화된 규격 자재와 다른 절단 작업에서 발생된 난척 자재가 있다. 주문서는 철근의 길이별로 요구되는 갯수를 담고 있다. 이 과정에서 생산되는 철근은 다음 가공 단계인 절곡 과정으로 제공되고, 재활용이 가능한 난척 자재는 다른 절단 작업이나 창고에 보관하게 된다. 이와 함께, 이 절단 공정에서 재활용이 되지 않는 남은 조각이 발생되게 되는데, 이 조각은 일정 크기 이하이면 더 이상 활용되지 못하고 고철로 버려지게 된다.

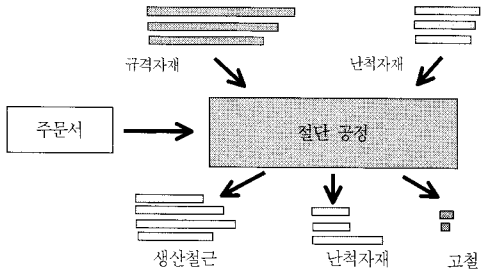


그림 1. 철근절단작업공정
Fig. 1 Process of Cutting Ironbar

## 2. 관련 연구

자재 절단 문제를 선형계획법으로 풀기 위해 Gilmore와 Gomory가 형식화한 수리적 모델링을 Kantrovich의 간략화된 표기법에 따라 표현하면 다음과 같다.[7] 주문 내역이 n개의 주문내역에서 $i$ 번째 부품의 길이 $w_i$의 주문량은 $d_i, i=1,...,n$로 표현되며, 자재는 고정된 크기 $W$로만 제공된다고 가정하고, 전체 공급량 $K$개에서 $k=1,...,K$의 범위안에서 $k$번째로 지정된 자재를 나타낸다. 이때, 자재 절단 문제의 목적은 다음 식과 같다. $x_{ik}$는 $k$번째 자재에 $i$ 번째 주문 부품이 얻어질 갯수를 나타내며, $y_k$는 $k$번째 자재가 사용되면 1, 아니면 0을 가진다.

$$Z^* = \min \sum_{k=1}^{K} y_k$$

$$\text{s.t.} \quad \sum_{k=1}^{K} x_{ik} \geq d_i \quad , i=1,...,n \qquad \langle \text{식 } 1 \rangle$$

$$\sum_{i=1}^{n} w_i x_{ik} \leq W y_k \quad , k=1,...,K \qquad \langle \text{식 } 2 \rangle$$

$$x_{ik} \in \{0,...,u_i\} \qquad \langle \text{식 } 3 \rangle$$

$$y_k \in \{0,1\}, \; k=1,...,K$$

$$\text{이때, } u_i = min\{d_i, \left\lfloor \frac{W}{w_i} \right\rfloor\} \qquad \langle \text{식 } 4 \rangle$$

〈식 1〉은 주문내역이 각각 만족되어야 할 조건을 뜻하며, 〈식 2〉, 〈식 3〉, 〈식 4〉는 배낭 문제에서 생성되는 절단 패턴 ($\overrightarrow{x_k} = [x_{1k},...,x_{nk}]$)이 적용 가능해야 함을 보장하는 제약조건이 된다.

이 논문에서 다루고 있는 철근 절단 문제는 자재가 하나로 고정된 것이 아니라 여러 개의 길이로 제공되므로, 위의 수리 모델에서 모든 자재에 대해 $W$로 표현하여 크기가 고정된 것을 $k$번째 자재에 대해 $W_k$로 표현하는 방식으로, 즉 각각의 자재에 대해 각각 다른 크기를 가진다는 것을 구분하여 표시하면 된다. 이런 경우에는 자재의 공급 갯수를 최소화시키는 것보다는 절단 작업으로 생겨나는 버려지는 조각(waste)의 량을 최소화시키는 것으로 최적화의 목적을 수립하는 것이 바람직하다.

자재 절단 문제를 선형계획법으로 풀기 위한 문제 모형의 수학적 표현은 간단하지만, 최적의 해를 구하는 과정은 알고리듬의 특성상 NP-hard 문제로 알려져 그 풀이 과정은 매우 복잡하고 느리다.[8] 더욱이 변수의 개수가 커지는 경우에는 해를 구하는 시간을 예측하기 어렵다. 이에 대한 대안으로 동적계획법을 사용하여 자재 절단 문제를 재구성하고, 보다 효율적인 계획 수립 알고리듬을 사용하고자 한다. 동적계획법을 자재 절단 문제에 응용한 사례로 Raffensperger[9]는 선형계획법과 동적계획법을 혼합하여, 최적화 문제의 주(master) 알고리듬은 선형계획법의 일종인 정수계획법으로 하고 부문제(subproblem)를 동적계획법으로 해결하는 부분적인 활용방법을 소개한 바 있다. 여기서는 해를 구하는 주(master) 알고리듬을 동적계획법으로 행하여 근사 최적의 해를 구하는 방안을 제시하고자 하는 것이므로, 전체 문제에 대한 동적 계획법을 위한 모델링이 필요하다.

# III. 알고리즘 설계

## 1. 절단 계획 알고리듬의 문제 모델링

자재 절단 문제를 동적계획법으로 풀어내는 것은, 기본적으로 다음과 같은 가정 하에서 이루어진다. 즉, 현재 상태에서 앞으로 취해질 최적 탐색 전략은 이전에 취해져서 얻어진 탐색결과와 무관하다고 가정한다. 그렇게 되면, 현재 단계에서 취해야 할 최적 탐색 전략을 현재 단계의 상태 정보에 의존하여 최선의 선택을 수행함으로써 전체적으로 최적의 탐색을 행할 수 있다. 자재 절단 문제는 이러한 가정이 성립되는 문제는 아니다. 하지만, 최적의 해를 찾기 보다는 근사최적의 해를 위한 효율적인 접근법의 하나로 동적 계획법을 사용하려고 하는 것이므로, 이 가정을 최대한으로 만족시킬 수 있는 조건으로 각 단계에서 진행하는 휴리스틱을 활용한 탐색 전략을 취하도록 한다.

동적 계획법을 위한 회귀식은 $F_k$와 $F_{k-1}$로 표현하며, $F_k$는 매 $k$단계별로 작업이 진행될 때 생겨나는 남는 조각(waste)의 합으로 정의하고, 이것을 최소화시키는 것을 최적화 과정의 목적으로 정의한다. 초기 상태 $S_0$에서는 초기값 0을 할당한다. 각 단계를 나타내는 단계변수는 $k$가 되며, 각 단계의 의미는 하나의 자재 단위로 절단할 수 있는 절단 패턴 $\overrightarrow{x_k}$을 적용하여 절단 작업을 할 수 있는 작업의 단위로 정의한다. 각 단계

의 상태를 나타내는 상태변수는 $S_k$, 각 단계에서 결정해야 될 결정변수는 $\overrightarrow{x_k}$와 $W_k$가 된다.

$$F_0(S_0) = 0, \quad \text{〈식 5〉}$$

$$\text{이때, } S_0 = [\sigma_0, \overrightarrow{\delta_0}] \quad \text{〈식 6〉}$$

$$\overrightarrow{\delta_0} = [\delta_{10}, ...\delta_{i0}, ..., \delta_{n0}]$$
$$where\ \delta_{i0} = d_i, i = 1, ...n$$
$$\sigma_0 = \{W_1, ..., W_j, .., W_m\}$$

$$F_k(S_k) = \min\{f_k(W_k, \overrightarrow{x_k}) + F_{k-1}(S_{k-1})\} \quad \text{〈식 7〉}$$

$$\text{이때, } f_k(W_k, \overrightarrow{x_k}) = (W_k - \sum_{i=1}^{n} w_i x_{ik}) \quad \text{〈식 8〉}$$

$$S_k = [\overrightarrow{\delta_k}, \sigma_k], \quad \text{〈식 9〉}$$

$$\overrightarrow{\delta_k} = [\delta_{1k}, ...\delta_{ik}, ..., \delta_{nk}]$$
$$where\ \delta_{ik} = \delta_{ik-1} - w_i x_{ik}, i = 1, ...n$$
$$\sigma_k = \sigma_{k-1} - \{W_k\}$$

〈식 5〉는 초기 상태의 식이며, 〈식 6〉의 상태 변수 $S_0$는 주문내역에 대한 상태 $\overrightarrow{\delta_0}$와 재고 상황을 나타낸 상태 $\sigma_0$로 이루어진다. 벡터변수 $\overrightarrow{\delta_0}$는 초기의 $n$개 종류의 주문 내역(demands)인 $d_i, i = 1, ..., n$ 값을 담고 있다. 집합변수 $\sigma_0$는 초기의 재고 상황으로서 전체 $m$개의 자재들을 각각 표현한 $W_j$를 원소로 하는 집합을 나타낸다. 〈식 7〉은 단계 $k$일 때, 이전 단계의 상태와 결과를 바탕으로 이 단계에서 결정되어야 할 조건을 회귀식으로 나타낸 것이다. 이 때, 단계 $k$의 $S_k$는 이 단계에서 처리되어야 할 남은 주문내역의 상태와 이 단계에서 현재 남아있는 자재의 현황 상태를 나타내는 것이 된다. 그리고, 〈식 8〉은 단계 $k$에서 절단 패턴 $x_k$를 취하여 절단작업을 행하는 경우에 버려지는 조각(waste)을 표현한 것이다. 이 단계에서의 작업으로 영향을 받는 상태변수 $S_k$의 변화는 〈식 9〉로 표현된다.

## 2. 절단 계획 알고리듬

앞서 언급한 동적계획법에 필요한 가정이 성립하면 동적계획법은 〈그림 2〉와 같은 해 공간에서 최적의 해를 찾아가는 것을 보장한다. 철근 절단 문제에서는 이러한 가정이 성립되지 않지만, 이를 무시하여도 최적의 해와 크게 멀어지지 않도록 탐색 영역을 제한하는 휴리스틱을 활용하여 $k$번째 단계마다 최선의 후보를 선택하여 전체적으로는 최적 해에 가까운 근사 해를 찾는다.

*〔휴리스틱 1〕 가장 큰 길이 주문내역을 우선 처리한다.*

첫 번째 휴리스틱은 단계 $k$를 진행하면서 $\overrightarrow{x_k}$를 생성시

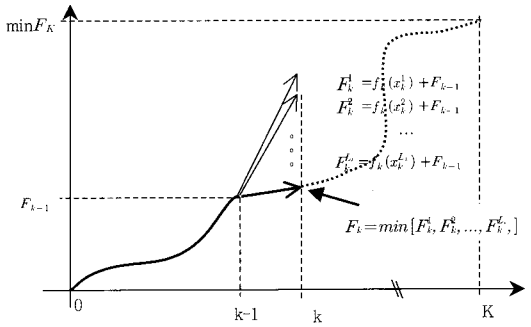


그림 2. 동적계획법의 해공간
Fig. 2 Solution Space of Dynamic Programming

키는 순서에 있어서, 남은 주문내역 $\overrightarrow{\delta_{k-1}}$에서 가장 큰 길이의 주문내역 $\delta_{max.k-1}$을 항상 포함시켜서 절단 패턴의 후보 $x_k^l$, $l = 1, ..., L_k$를 생성하는 것이다. 이 휴리스틱은 가방에 짐을 꾸릴 때, 가장 큰 물건을 먼저 넣고 나머지 공간에 남은 작은 물건들을 채워 나가는 것과 같은 원리이다.

*〔휴리스틱 2〕 유효한 패턴을 생성한 후, 이에 가장 적합한 재고(stock)의 짝을 배정한다.*

두 번째 휴리스틱은 단계 $k$를 진행하면서 각 단계에서 적용이 가능한 유용한 절단 패턴의 후보들 $x_k^l$, $l = 1, ..., L_k$에 대해, 각 단계에서 제공이 가능한 재고 집합 $\sigma_{k-1}$에서 가장 적합한 $W_k^t$을 찾아서, 절단 패턴 후보 $x_k^l$에 대해 가장 적합한 $W_k^t$를 지정한다. 이것은 짐을 꾸릴 가방의 크기가 다양하게 제공된다고 할 때, 꾸릴 짐의 조합 패턴과 이에 적합한 가방을 선택하여 짝을 이루게 하는 것과 같다.

이 알고리듬에서 필요한 절단 패턴의 생성과정은 너비우선 탐색방법(breath-first search)를 사용하여 절단 패턴의 조합을 생성한다. 이 과정에서 무작위로 모든 가능한 조합을 고려한다면 탐색공간의 크기가 매우 커진다. 그러나, 모든 조합의 패턴이 유효한 것이 아니므로 모든 가능한 패턴을 생성할 필요는 없으므로, 다음과 같은 휴리스틱을 활용하여 패턴의 생성과정에서 유효하지 않은 패턴을 걸러내고, 정해진 유한 개수의 범위 내에서 패턴의 후보를 생성하도록 하였다.

*〔휴리스틱 3〕 절단 패턴의 중복 방지와 최대 길이 제한에 따라 유효한 패턴을 생성한다.*

중복된 패턴을 방지하기 위한 규칙성으로, 너비우선 탐색과정에서 새로운 주문 분절을 추가하여 탐색공간의 깊이를 확장할 때, 이전 깊이에서 마지막에 추가된 분절의 크기보다

큰 주문 분절은 고려하지 않아도 된다. 그리고, 재고의 최대 길이가 12000mm로 제한된다. 그러므로 실제적인 탐색공간이 크게 줄어든다.

Algorithm Cutting-Ironbar Planning
$S=\{W_1,\ldots,W_m\}$ : 재고집합,
//이때, $W_i$은 $i$번째 재고의 길이
$\vec{w}=[w_1,\ldots,w_n]$ : 주문의 길이벡터,
//이때, $w_i$는 $i$번째 품목의 길이
$\vec{d}=[d_1,\ldots,d_n]$ : 주문의 갯수벡터,
//이때, $d_i$는 $i$번째 품목의 갯수
// $\vec{w}$와 $\vec{d}$는 길이에 따라 미리 순서화됨.
Begin
$F_0=0$, $S_0=S$, $\vec{d}_0=\vec{d}$, $i_{max}$=1, k=1
repeat
// $i_{max}$가 포함된 절단 패턴을 생성
$X_k$ = generate_patterns($i_{max}$, $\vec{d}_k$, $S_k$);
repeat
// 최소의 Waste를 발생시키는
// 절단 패턴과 재고의 쌍을 찾는다.
$f_k=\min_{l=1}^{L_k}\min_{j=1}^{m}\left\{W_j-\sum_{i=1}^{n}w_i x_{ik}^{l}\right\}$
// 이때, $\vec{x}_k$와 $W_k$가 결정된다.
$F_k=f_k+F_{k-1}$
$\vec{d}_k=\vec{d}_{k-1}-\vec{x}_k$
$S_k=S_{k-1}-\{W_k\}$
k = k+1
until ( $d_{i_{max}k-1}\le 0$ )
$i_{max}=i_{max}+1$
until ( $i_{max}>n$ OR $\sum_{i=1}^{n}d_{ik-1}=0$ )
End Begin

그림 3. 절단 계획 알고리듬
Fig. 3 Algorithm for Cutting-Ironbar Planning

*〔휴리스틱 4〕 후보 절단 패턴의 최대 개수를 제한한다.*

단계 $k$가 진행될수록 주문내역의 종류가 줄어서 후보패턴의 이론적인 탐색공간은 줄어든다. 하지만, 주문내역의 종류가 많으면서 작은 길이들로 주어진 경우에는 휴리스틱 3으로도 탐색공간이 그리 크게 줄어들지 않는다. 그러므로, 알고리듬의 효율성을 위해서 후보 패턴의 최대 개수를 제한할 필요가 있다. 이렇게 탐색공간의 크기를 제한하더라도 결과는 큰 차이를 보이지 않을 것으로 예상된다. 이것은 가방에 짐을 꾸릴 때, 물품의 크기가 가방의 크기에 비해 작은 경우에 무작위로 짐을 꾸리더라도 남는 공간은 그리 큰 차이를 보이지 않는다는 것과 같다.

이러한 휴리스틱을 적용하여 동적계획법에 근거한 절단계획 알고리듬의 전체적인 순서도는 〈그림 3〉과 같다.

## IV. 시스템 설계

### 1. 시스템 구조도

개발하고자 하는 소프트웨어는 철근절단 작업공정의 계획을 컴퓨터 시스템에 의해 자동으로 수립해주는 프로그램이다. 이러한 철근 절단 계획 소프트웨어의 구조도는 〈그림 4〉와 같다.

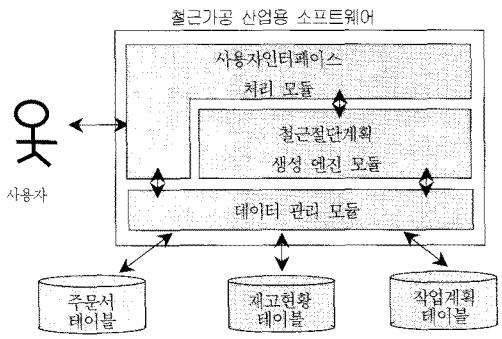


그림 4. 소프트웨어 구조도
Fig. 4. Software Framework

작업 계획 수립이 요구되는 주문서가 입력되면 이를 주문서 테이블에 저장하고, 투입 철근의 재고량은 재고현황 테이블에 저장되어 있다. 사용자가 특정 주문서에 대한 철근 절단 계획을 수립하려면, 철근 절단 계획 생성 엔진에 계획 수립 작업을 요청하여 작업 계획을 생성하고 작업 계획 테이블에 저장하도록 한다. 계획 생성 엔진은 근사 최적의 해를 산출하는 절단 계획 수립 알고리듬을 실행하는 독립적인 모듈이 되며, 주문서 테이블과 재고 현황 테이블, 작업계획 테이블의 데이터베이스 관리를 담당하는 데이터 관리 모듈은 사용자 인터페이스 모듈과 연결되어 동작함으로써 사용자가 주문서와 재고현황을 검토하거나 수정과 편집을 할 수 있도록 한다. 계획 생성 엔진 모듈과 연결 동작하여 계획 수립에 필요한 데이터를 공급하고, 최적의 작업계획을 생성하도록 한다.

### 2. GUI 설계

태스크 기반의 GUI 모델링 방법론을 사용하여 화면 구성과 GUI 구성요소의 설계가 이루어졌으며, 빠른 프로토타입 구현과 사용성 테스트를 거친 후[11], 이 논문에서는 2번째

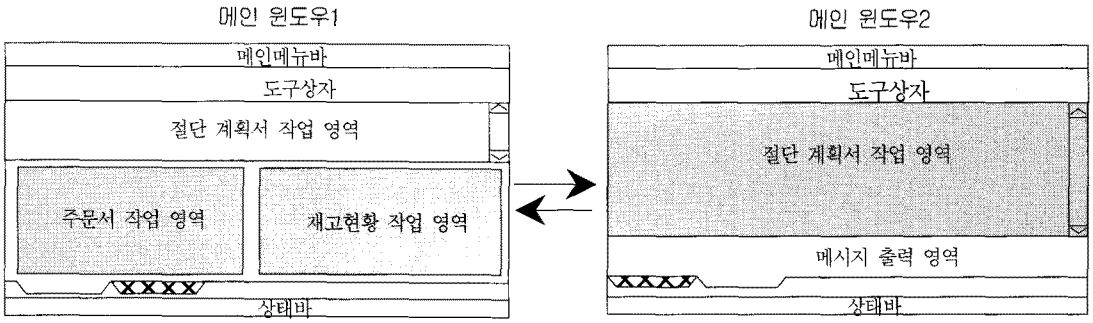


그림 5. 메인화면 구성의 설계
Fig. 5. GUI Design of Main Windows

프로토타입을 재설계하였다. 〈그림 5〉에서 보이듯이, 최상위의 주문서 작업, 재고현황 작업, 절단계획 수립 작업을 메인 윈도우1의 화면 레이아웃에 할당하고, 추가적으로 메인 메뉴바와 도구상자, 상태바, 등의 구성요소를 가지고 있다. 주문서 작업과 재고현황 작업이 성공적으로 완료되면, 절단 계획서 작업이 가능해지며, 사용자가 절단 계획 생성을 요청하게 되면 절단 계획 생성 후에는 메인윈도우2로 화면이 전환된다. 이때, 절단계획서 작업영역에는 산출된 절단 계획서를 출력하게 되고, 그 아래 영역에는 주문서 작업과 재고현황 작업 대신에 절단 계획 생성 과정에서 출력되는 메시지를 보이는 메시지 출력영역을 갖는다. 두개의 윈도우는 아래쪽에 위치한 탭 선택 버튼으로 수동 전환이 되도록 한다.

### 3. GTK+ 개발 환경

공개 소프트웨어 개발 툴킷인 GTK+는 GNU LGPL 라이센스 정책을 따르는 크로스-플랫폼 GUI 툴킷이다.[12][13] 전세계 언어가 사용가능한 다국어 지원과 다중 프로그래밍 언어을 위한 언어 연동부(language bindings)가 풍부하게 지원 가능한 장점과 함께, GLADE와 같은 RAD(Rapid Application Development) 도구가 무료로 제공되는 점도 GTK+가 널리 사용되는 이유이다.[14]

## V. 구현결과

### 1. 프로토타입 구현 결과

프로토타입 구현은 LINUX 플랫폼의 GNOME 데스크탑 환경에서 C 언어로 개발되었으며, GNU C 컴파일러를 이용하였다. 또한 Windows XP급 이상의 플랫폼에서도 MinGW GNU C 컴파일러와 MSYS 쉘 에뮬레이터를 이용하여 다시 컴파일하여 이식 가능함을 확인하였다.

화면 레이아웃 설계에 맞춰서 메인 윈도우와 두개의 보조 대화창의 실제 구현 화면은 〈그림 6〉와 같다. 초기 화면은 메인 윈도우 하나만 비어있는 상태로 화면에 표시된다.

### 2. 프로토타입 평가 결과

#### 2.1 알고리듬의 성능 분석

동적계획법에 의한 자재 절단 계획 알고리듬의 성능 실험을 위해 〈표 1〉과 〈표 2〉에서 주어진 데이터를 주문 내역과 자재 내역의 실험 데이타로 사용하였다.

표 1. 주문 내역
Table 1. Order List

| 유형 | 길이(mm) | 생산량 |
|---|---|---|
| D10 | 4500 | 97 |
| D10 | 3600 | 610 |
| D10 | 3100 | 395 |
| D10 | 1400 | 211 |

표 2. 자재 내역
Table 2. Stock List

| 유형 | 길이(mm) | 재고량 | 유형 | 길이(mm) | 재고량 |
|---|---|---|---|---|---|
| D10 | 12000 | 80 | D25 | 12000 | 50 |
| D10 | 11000 | 90 | D25 | 11000 | 50 |
| D10 | 10000 | 100 | D25 | 10000 | 80 |
| D10 | 9000 | 90 | D25 | 9300 | 75 |
| D10 | 8000 | 80 | D25 | 8300 | 35 |
| D10 | 7000 | 50 | D25 | 7000 | 35 |
| D10 | 6000 | 30 | D25 | 6000 | 25 |
| D10 | 9250 | 8 | D25 | 7450 | 5 |
| D10 | 3380 | 5 | D25 | 620 | 3 |

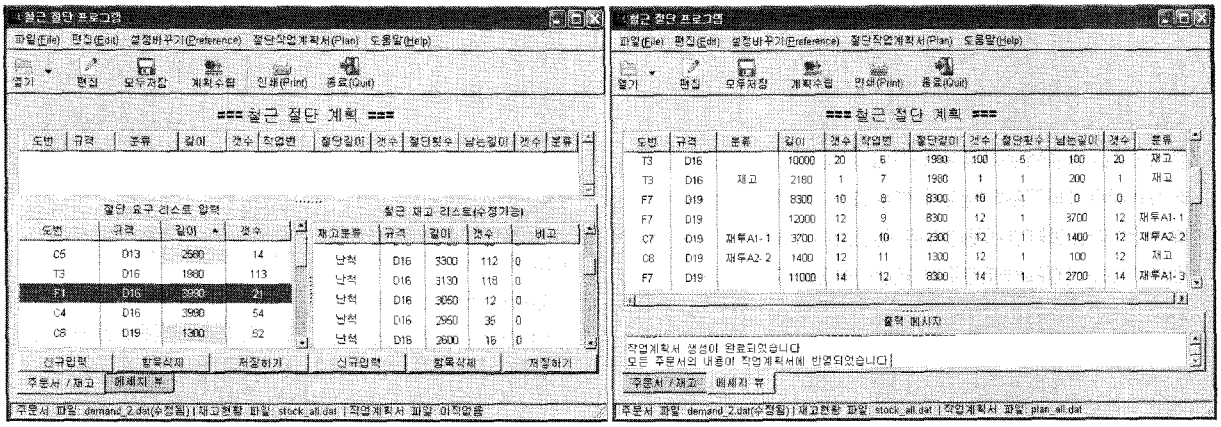

그림 6. 메인 윈도우의 화면 레이아웃
Fig. 6. Layout of Main Windows

〈표 3〉에서 조건 1은 자재가 표준 규격의 다양한 크기로 무제한 공급이 가능하다고 할 때, 동적계획법에 의한 결과를 보이고 있다. 이 경우에는 동적계획법에 의해서 나온 절단 계획이더라도 매우 높은 수율이 얻어지는 것을 확인할 수 있다. 하지만, 현실적으로는 창고에 유한 갯수로 보관된 자재로 계획을 세워야 한다. 조건 2는 실제로 제한된 자재 현황에서의 결과를 보이고 있다. 실험 결과에서 보면, 동적계획법은 최적의 해는 아니지만 휴리스틱1과 휴리스틱 2를 적용함으로써 근사 최적의 해를 효율적으로 산출할 수 있음을 확인할 수 있다. 수작업으로 행하던 결과의 수율에 비하면 최적의 해는 아니지만 매우 유용한 결과라고 할 수 있다.

표 3. 철근절단 계획 알고리듬의 성능 실험
Table 3. Performance of Cutting-Iron Bar Planning

| 실험조건 | 자재 투입량 | 주문 생산량 | 수율 (%) |
|---|---|---|---|
| 조건 1: 동적계획법을 사용하고, 표준 규격별로 재고량이 충분히 제공되는 경우의 결과 | 419개 4271.0m | 1313개 4152.4m | 97.2 % |
| 조건 2: 동적계획법을 사용하고, 실제로 재고량이 한정 갯수로 제공되는 경우의 결과 | 470개 4544.5m | 1313개 4152.4m | 91.4 % |

### 2.2 프로토타입의 사용성 평가

일반적으로 사용자 인터페이스의 설계안에 대한 사용성 평가는 설계 원칙을 잘 적용하였는지에 대한 정성적인 측면에서 전문 사용자와 일반 사용자의 평가를 행하였다. 좋은 사용자 인터페이스 설계 원칙으로 여러 요소들이 있지만, 여기서는 가장 공통적으로 언급되는 5가지 설계 원칙[12]에 대하여 정성적인 평가를 행하였다.

표 4. 정성적인 사용성 평가
Table 4. Usability Test

전문사용자 평가:●, 일반사용자 평가:○

| 평가된 설계 원칙 | 상 | 중 | 하 |
|---|---|---|---|
| 1. 룩앤필(Look and Feel) | | ◉ | |
| 2. 피드백(Feedback) | ○ | ● | |
| 3. 일관성(Consistency) | ◉ | | |
| 4. 단순성(Simplicity) | ◉ | | |
| 5. 관대함(Generosity) | | ● | ○ |

평가 결과, 〈표 4〉에서 보인 것과 같이, 룩앤필, 피드백과 일관성, 단순성의 측면에서는 중상의 결과를 얻었으나, 관대함의 측면에서 중하의 결과를 얻었다. 관대함의 측면에서 불편한 사용성의 원인은 입력창의 작업을 행한 후, 실수로 창을 강제 종료하는 사용자의 실수에 대한 복구 기능(UNDO)이 미비한 점이다. 업그레이드 버전에서는 이러한 문제점들이 해소하도록 분석 내용이 반영되어야 하겠다.

## VI. 결론

이 논문은 철근 가공 산업 현장에서 스프레드쉬트에 의해 수작업으로 행하던 절단 작업의 계획 수립을 자동화된 시스템

에 의해 최적의 절단 계획을 생성하도록 하는 소프트웨어의 개발을 다루었다. 철근 가공 산업용 소프트웨어는 생산 자동화 소프트웨어의 일종으로, 수작업으로 진행되던 철근 절단 계획 수립의 업무를 전산화하여 최적화된 작업계획을 생성하는 자동 절단계획 수립을 핵심 기능으로 수행하고, 주문 및 재고 현황, 등 제반 자료에 대한 정보 관리 서비스 시스템과 연계도 이루어져야 하는 요구사항을 갖는다. 이를 위하여, 먼저 다중 규격의 1차원 자재 절단 문제에 동적계획법을 적용하고, 유한 범위의 조합 열에서도 근사 최적의 해를 찾을 수 있는 탐색 기법을 사용한 자재 절단 계획 알고리듬을 사용하였다. 그리고, 자동화된 철근 가공 산업용 소프트웨어는 작업 환경에 맞게 사용이 편리한 그래픽 화면과 사용자 인터페이스가 요구되는데, 공개 소프트웨어를 활용한 GUI 라이브러리 툴킷인 GTK+를 활용하여 이를 구현하였다.

개발된 소프트웨어는 철근 가공 산업의 현장 지식과 첨단 정보기술이 융합되어 얻어지는 결과물로써 철강 산업 분야에서 전통적인 방식으로 운영되고 있는 많은 소규모 철근 가공 공장에서 활용될 것으로 기대된다.

## 참고문헌


[1] Gilmore, P.C. and R. E. Gomory, *"A Liniear Programming Approach to the Cutting-Stock Problem,"* Operations Research, Vol. 9, pp. 849-859, 1961.

[2] 김상열, 박순달, 자재절단문제에서 패턴의 순서화 방법에 대한 연구, 한국경영과학회 '95 추계 학술대회 논문집, 406~419쪽, 1995년 9월.

[3] Haessler, R. W., *"Selection and Design of Heuristic Procedure for Solving Roll Trim Problem,"* Management Science, Vol. 34, No. 12, pp. 1460-1471, 1998.

[4] Wang P. Y., *"Two Algorithms for Constrained Two-Dimensional Cutting Stock Problems,"* Operations Research, Vol. 31, No. 3, pp. 573-586, 1983.

[5] 김상열, 박순달, 효율적인 2차원 길로틴 평면 절단 방법, 산업공학 논문지, 제8권 제2호, 151-159쪽, 1995년 7월.

[6] 김성훈, 최창훈, 철근절단계획 알고리듬의 설계, 디자인연구 논문집, 제5호, 83-96쪽, 상주대학교 산업디자인연구소, 2007년 5월.

[7] Kantorovich, L. V., *"Mathematical methods of organizing and planning production,"* Management Science, Vol. 6, pp. 363-22, 1960.

[8] Hu, T.C., *Combinatorial Algorithms*, Addison Wesley Longman Press, 1981.

[9] Raffensperger, John F. *"The marriage of dynamic programming and integer programming,"* Procs. ORSNZ 34th Annual Conference, Waikato, pp.49~58, 1999.

[10] 김상길, 김성훈, 박충식, 김재희, 구조적 기술에 의한 전문가 시스템의 사용자 인터페이스 개발 방법, 전자공학회지 제32권 B편, 제1호, 161-170쪽, 대한전자공학회, 1995년 1월.

[11] 김성훈, 철근절단계획 소프트웨어의 그래픽 사용자 인터페이스 설계, 디자인연구 논문집, 제6호, 73-84쪽, 경북대학교 상주캠퍼스산업디자인연구소, 2008년 5월.

[12] Andrew Krause, *Foundations of GTK+ Development*, Apress Pub., 2007.

[13] GTK+ Team, GTK+ Project homepage, *http://www.gtk.org*, 2008.

[14] Glade, Glade Project homepage, *http://glade.gnome.org*, 2008.


## 저 자 소 개


**김 성 훈**

1996년 2월 : 연세대학교 대학원 전자공학과 (공학박사)

1996년 3월~2006년 2월 : 영동대학교 컴퓨터공학과 부교수

2006년 3월~ 2008년 2월: 상주대학교 소프트웨어공학과 조교수

2008년 3월~현재 경북대학교 컴퓨터정보학부 조교수

〈관심분야〉 인공지능, 패턴인식, 생체인식, 지능형콘텐츠