냉연 공정에서의 작업단위 편성†

전치혁* · 이승만* · 박철순* · 강상엽* · 장수영* · 최인준* · 강전태**

A Job Sequencing Model for Cold Coil Production Processes†

C.H.Jun*, S.M.Lee*, C.S.Park*, S.Y.Kang*, S.Y.Chang*, I.J.Choi*, and J.T.Kang**

Abstract

A job sequencing model is developed and its computer system is tested for processing cold-rolled coils in Tandem Cold Mills(TCM) at the Pohang Iron and Steel Company. Given coils waiting to be processed, this system generates a sequence of jobs satisfying operational constraints for the TCM process. We formulate the problem as a constraint satisfaction problem and employ the back-tracking technique combined with looking ahead features in order to generate a feasible solution within a reasonable time. Our system is implemented in C language on 80486-based IBM PC. Some tests based on the real data show that our system is adequate with respect to search time and that it consistantly generates a good feasible solution.

1. 서 론

철강공정은 제선공정, 제강공정, 열연공정, 그리고 냉연공정이 있으며, 냉연공정에서는 냉간압연에 의해 냉연강판, 표면처리강판, BP(Black Plate)등의 냉연제품을 생산한다. 포항제철㈜의 냉연 공정흐름은 그림 1과 같다.

그림 1에서 보듯이 어떤 냉연제품을 생산하기 위해서는 산세(pickling;PL) 공정과 냉간압연(Tandem Cold Mills; TCM) 공정을 거쳐야 한다. 이들 공정에 대한 자세한 설명은 참고문헌 [1], [2]에 나와 있다.

본 연구에서는 냉간압연공정을 중심으로 작업단위편성에 있어서의 여러 문제점을 기술하고 이를 해결하는 방법을 찾고자 한다. 작업단위편성이란 하루 단위로 작업하여야 할 대상재를 선정하여 공정의 제약조건을 만족하는 이들의 순서를

† 본 연구는 산업과학기술연구소를 통한 포항제철㈜ 계약과제에 의해 지원되었음.

* 포항 공과대학 산업공학과

** 포항제철㈜ 생산관리부

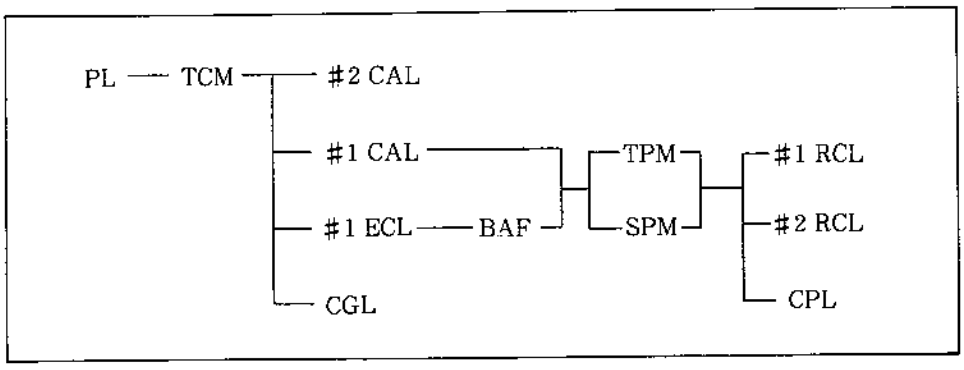


그림 1. 냉연 공장의 공정흐름도

결정하는 것을 말한다. 이러한 작업단위편성 문제는 그림 1의 모든 공정에서 갖고 있다. 작업대상재는 작업을 대기하고 있는 소재와 데이타베이스로 부터 전후 공정의 물류 밸런스, 설비조건 등을 고려하여 선정되는데, 본 논문의 분석에서는 작업대상재가 주어진 것으로 가정하여 이들 중 해당 공정의 작업제약조건을 만족시키는 일련의 작업재를 선택하고 순서를 결정하는 것으로 문제를 국한한다.

실제 현장에서는 이러한 작업단위편성을 단순한 정렬기법에 의존하여 처리하고 있다. 즉, 여러 다른 제약조건들 중 최우선 기준을 선정하고 이를 이용하여 정렬하고, 다시 그 결과를 다음 우선순위의 기준으로 정렬하는 과정을 반복한다. 그러나, 편성기준이 복잡해 짐에 따라 여러가지 문제점이 대두되고 있다. 그중 하나로 상호보완적, 배타적으로 발생하는 기준항목에 대한 고려를 단순한 정렬기법에 의해서는 처리할 수 없으므로 단위편성이 불가능하게 되는 경우가 발생한다. 또한 현실적으로는 기준항목을 엄격히 준수하지 않는 것을 허락할 수 있기 때문에 이를 활용하는 것이 불가피할 때가 있으나, 단순한 정렬에서는 이를 고려하지 못하게 된다. 또한, 단순한 우선순위에 입각한 정렬은 모든제약조건을 만족 시키지 못하는 경우가 종종 있으므로 사용 가능하지 못한 일정계획을 주게 되어, 많은 수작업을통하여 중요하지 않은 제약조건부터 이완시키는 과정(이 경우 생산제품의 품질 또는 작업효율에 문제가 있을 수 있다.)을 숙련된 일정계획 전문가의 지식에 입각하여 수행해야 하므로 많은 시간이 소요되며, 결국에는 정렬한 결과를 모두 고쳐야 하는 결과가 초래 되기도 한다. 한편, 간단한 논리의 정렬과정을 거친후 제약조건이 만족되는 묶음을 하나의 단위로 임의적으로 처리하는 방법이 가능하기는 하나, 이에 따르는 문제는 부정기 적인 Roll의 교체 횟수로 나타난다.

즉, Roll교체가 잦아짐에 따라, 생산성의 저하와 원가요인 상승의 결과를 초래하고 있으며 무리한 일정계획을 시행할 때 오는 품질의 저하도 우려되고 있다.

따라서, 이러한 문제를 해결하는 길은 이러한 작업단위 편성문제를 제약 만족 문제(Constraint Satisfaction Problem;CSP)[3]로 모형화하여 신속히 제약조건을 만족하는 하나의 작업단위를 찾아 주는 것만이 적절한 접근 방법이라고 보았다. 즉 목적식이 있는 일반적인 일정계획 문제의 유형으로 접근하지 않고, 단지 제약조건을 만족하는 하나의 가능한(Feasible) 일정계획을 신속히 찾아주는 것을 목표로 했다. 다만, Roll의 교체 횟수를 최소화 하기 위하여, 작업단위내의 크기, 즉 총단중(중량)을 최대화 하는 것을 목표로 할 수 있으나, 작업단위의 총단중을 만족할 수 있는

크기로 미리 정하여, 이를 하나의 제약조건으로 처리할 수 있으므로 주어진 총단중 제약조건을 만족하는 하나의 가능해를 신속히 찾는 것으로 충분하다고 판단 되었다. 이에따라 본 연구에서는 제약 만족 문제에서 사용하는 기법인 Back-tracking 및 Look-Ahead 기법[4][5]을 적절히 응용하여 하나의 가능한 작업단위편성을 신속히 구하는 시스템을 개발하였다.

여기서, 일정계획에서 일반적으로 사용되는 혼합정수계획법과 같은 최적화 모델을 설정하지 않은 이유는 혼합정수계획 모형을 도출할 경우, 너무 많은 정수변수를 필요로 하게 되며, 문제의 특성상 제약조건이 반드시 준수되지 않아도 되는 부분이 많기 때문이다. 또한 목적식의 설정에 있어, Roll의 교체회수 혹은 단위별 총단중, 등 여러가지 요건이 고려되야 하는 어려움이 있다. 특히, 실제 현상에서는 제약조건을 만족하는 어떤 편성안들의 상호 우월성을 판단하기가 어려운 실정이다. 이에따라 스케쥴 편성에 있어 수량적으로 목적함수를 부여함이 곤란하다.

2. 냉간압연공정의 제약조건

본 절에서는 TCM공정의 제약조건에 대한 설명을 하기로 한다. 그러나 대상공정의 현장기술에 대한 고유권한을 보호하기 위하여 문맥의 명확성을 잃지 않는 범위에서 정확한 값 대신 '어느 정도', '주어진' 등의 용어를 사용하였다. 냉간압연(TCM)은 산세가 끝난 코일을 상온에서 0.18㎜－1.6㎜의 제품두께로 압연하는 작업을 말한다. TCM공정에서는 극박, 중박, 후물, 전강재로 코일들을 분류하며(분류된 각각을 단위라 함), 각 단위별로 작업단위를 편성해야 한다. 이러한 작업단위를 편성하기 위해서는 지시대상의 여러 냉연코일들의 작업순서를 단위 편성 기준에 부합되도록 결정하여야 한다. 여기서, 모든 제약조건들은 동시에 만족되어야 하나 제약조건들간의 상관관계는 없으며, 편성기준간에는 우선순위가 존재한다. 본 공정에서 만족되어야 하는 제약조건들을 살펴보면 다음과 같다.

1) 작업단위 내에서 코일의 폭은 광폭에서 협폭순으로 이루어져야 하며, 전후 코일의 폭 편차가 주어진 기준이내인 경우에는 동일폭으로 간주한다.(따라서, 다른 제약조건에 의해 폭순에서의 역전이 일어날 수도 있다.) 그리고, 동일 작업단위 내에서 처음 코일과 마지막 코일의 폭 편차는 또한 다른 기준이내에 있어야 한다.

2) 코일의 두께는 X-Ray Set치를 기준으로 하는데, 동일폭 하에서 후물에서 박물순으로 작업순서를 결정해 주어야만 하며, 만약 폭이 변경되었을 경우에는 X-Ray치 추이의 평활화(smoothing)를 위하여 전 폭의 마지막 코일의 X-Ray치와 편차가 가장 작은 X-Ray치를 갖는 코일을 처음으로 하여 후물에서 박물순 또는 박물에서 후물순으로 작업순서를 결정해 주어야 한다. 그리고, 동일 작업단위 내와 작업단위 변경시 모두 전후 코일의 두께 편차는 열연 코일의 두께를 기준으로 어느정도의 범위이내에 있어야만 한다.

3) Roll의 교체주기를 감안하여 적정 작업단위 편성량을 중물과 후물인 경우에는 일정량에 가깝도록 만들어 주어야 하며, 극박과 전강인 경우에는 또다른 주어진 양에 가깝도록 만들어야 한다.

4) 두께 추이의 평활화를 위하여 전 X-Ray치의 마지막 두께와 차가 가장 작은 두께를 처음으로 하여 작업순서를 다시 결정해 준다. 만일 아직도 동일조건인 코일이 존재한다면 탄소 함유량에 따라서 고탄소재에서 저탄소재 순으로, 납기일을 비교하여 선납기에서 후납기순으로, 그리고 생산 번호순으로 편성한다.

3. Constraint Logic Programming을 사용한 프로토타입 개발

이 과정의 목적은 작업단위편성 문제를 분석하고 스케쥴링의 논리를 설계함과 동시에 빠른 시간내에 프로그래밍하여 실행해 봄으로써 시스템 설계의 타당성을 검토하며, 아울러 주 프로그램으로의 전환이 용이한 프로토타입의 개발에 있다. 프로토타입은 CLP(R)을 사용하여 개발되었다.

3.1 CLP(R)(Constraint Logic Programming, Real Domain)

논리 프로그래밍(Logic Programming)은 프로그래머가 tree search의 과정을 언급하지 않고 문제의 논리적관계(semantic relation)만을 선언하면 내장된 backtracking기능으로 해를 찾아준다. 따라서 매우 간결한 선언만으로 프로그래밍을 할 수 있는 장점이 있다[6]. 그러나 문제의 복잡성이 증가하게 되면 탐색영역이 기하급수적으로 증가하게 되므로 논리 프로그래밍만으로 그 효율성을 잃게 된다. 이 문제를 해결하기 위해서 제약조건 만족검사(Constraint Satisfaction 혹은 Consistency Checking)기법을 논리 프로그래밍 내에 접목시키는 개념이 제시되었으며, 이것에 의해 탄생한 프로그램 언어 중의 하나가 CLP(R)이다.[7, 8]

CLP (R) (Experimental Implementation of Constraint Logic Programming Paradigm in the Domain of Real Arithmetic)은 Prolog의 상위언어로 기존 Prolog의 기능을 내장하고 있다. 그 외에 기존의 논리 프로그래밍에서 다루지 못했던 다음과 같은 특징을 가지고 있다. 실수의 표현 및 계산기능, 입력과 결과의 출력을 제약조건의 형태로 내재적(implicit)으로 표현할 수 있는 기능, 추론기관(inference engine)외에 계산기관(constraint solver)을 내장하고 있어서 필요에 따라서 Gaussian Method에 의해서 제약조건내의 미결정 변수값들을 계산해 주는 기능등이 있다.[9] 이런 기능들은 기존의 논리 프로그래밍의 한계를 극복하게 해 주었고 결과적으로 논리 프로그래밍의 장점인 높은 표현력과 취약점인 계산능력 모두를 강화시킴으로서 CLP(R)이 다수의 실제적인 응용, 예를 들면, 전기회로 설계, 재무관리분석 전문가 시스템, Discrete Event Processes의 인식 등의 다양한 부분에 사용 될 수 있게 하였다.[10, 11, 12]

3.2 프로토타입 개발과정 및 타당성 검토

1) 제약조건 검사

논리 프로그래밍으로는 다음의 예와 같이 전후코일 간의 제약조건을 전코일의 속성값을 대하여 병합가능한 후코일의 속성을 논리적인 관계로서 선언적으로 표현하기가 용이하다. 예 1은 TCM 공정에서 전후코일의 폭차가 3㎜이내일 때 동일폭으로 간주하도록하는 제약조건을 표시한 예이다.

예 1

```
same_pok(Previous_Pok, Current_Pok) : -
        abs(Previous_Pok-Current_Pok) = <3.
```

이와 같은 방법으로 제약조건을 표현함으로써 다양한 제약조건을 쉽게 정의할 수 있으며, 제약조건의 추가, 변경이 용이하다. 이러한 관계들이 모든 제약조건에 대해 선언되면, 작업편성은 이와같은 제약조건들을 모두 만족하는지를 검사하는 과정을 거쳐서 코일단위로 이루어지며, 이 과정을 반복하여 작업단위가 완성된다.

2) Backtracking

단위편성을 위해서 전 코일에 병합할 수 있는 하나의 후코일을 찾기 위해서는 최악의 경우, 모

든 대상 코일들에 대해서 제약조건을 만족하는지를 검사해야 한다. 이때 코일 각각은 제약조건을 만족할 수도 있고 아닐수도 있다. 만족하지 않을 경우에는 만족하는 코일이 발견될 때까지 모든 대상 코일에 대해 검사를 계속해야 한다. 이 때, 논리 프로그래밍 기법을 이용하면 이 검사 과정의 알고리듬을 직접 코딩해야 하는 수고를 덜 수 있다. 이것은 논리 프로그램에서는 다음의 예와 같이 논리적 관계만을 선언해주면 이 관계를 만족하는 스케쥴(sequence of coils)을 찾을 때까지 프로그램 스스로 Backtracking을 반복한다.

예 2

```
schedule(Current_schedule, Result_schedule) : -
    get_a_coil(Next_Coil),
    not look_ahead_failure(Current_schedule,
    Next_Coil, Remained_Coils),
    test_constraint_satisfaction(Current_sched-
    ule, Next_Coil),
    update_schedule(Current_schedule, Next_Coil,
    Updated_schedule),
    not test_end_of_schedule(Updated schedule),
    schedule (Updated_schedule, Result_schedule).
```

예 2를 보면 대상재 중에서 하나의 코일을 선택(get_a_coil)한 후 이 코일이 스케쥴로의 첨가 가능여부를 검사한다(test_constraint_satisfaction). 이때 검사가 실패할 경우에는 또 다른 코일을 선택하도록 다시 get_a_coil으로 backtracking해서 또 다른 코일에 대해서 검사를 반복한다(reinstantiation). 이런식으로 제약조건이 만족되는 코일을 발견할 때까지 모든 코일을 검사하게 된다. 이와같이, 프로그램 상에서는 backtracking에 대한 구체적 방법론을 언급할 필요가 없이 자동적으로 수행되므로 프로그래머의 부담과 코딩 시간이 크게 줄어들게 된다. 아울러 프로그램 자체가 스케쥴링의 논리를 그대로 표현하고 있기 때문에 프로그램 자체가 일종의 실행가능한 명세언어(Executable Specification Language)라고 볼 수 있다.

3) Grouping과 Look Ahead

앞서의 backtracking 방법의 가장 큰 문제점은 최악의 경우 탐색과정에서 모든 경우의 수만큼 검색이 이루어진다는 점이다. 따라서, 실행시간을 줄이기 위해서는 가급적 불필요한 검색 범위를 줄어야 한다. 이를 위해 다음의 두가지 방법이 적용되었다. 첫번째는 코일 성격이 동일한 것들은 하나의 블럭으로 묶어 하나의 코일 단위로 처리하였다. 두번째는 CSP기법의 하나인 look ahead의 적용이다. CSP는 성격상 하나의 최적해를 구하기 어려울때 하나이상의 가능해를 빠른 시간에 구한다는 개념이다. 따라서 문제의 성격에 따라 가능해를 구해나가는 나름대로의 휴리스틱이 존재하게 된다. 본 연구의 경우에는 look ahead를 위해 다음과 같은 휴리스틱을 사용하였다.

〈Look Ahead 제약조건 1〉
Sum_of Scheduled_Weight_Until_Now
+Estimated_Sum_of_Weight_From_Now
>=Lower_Bound_for_Unit_Schedule

〈Look Ahead 제약조건 2〉
Pok_of_Fisrt_Coil-Pok_of_Current_Coil<=200

크게 위의 두가지 제약조건이 look ahead제약조건으로 사용되어 각 코일 스케쥴 될 때마다 검사되도록 하였다.

첫번째 제약조건은 현재까지 스케쥴된 단중의 합과 앞으로 스케쥴 가능한 단중들의 예상합을 더했을때, 그 합이 최소 적정 작업단위 편성량보다는 커야 앞으로 스케쥴을 계속했을 때 가능해가 나올 가망이 있다는 것을 나타내며, 두번째 제약조건은 현 작업단위의 첫번째 블럭과 접합가

능한 블럭중에서 가장 큰 폭을 지닌 코일간의 촉차가 주어진 범위(위의 예에서는 200㎜)이상 날 경우에는 현재의 스케쥴로부터는 더 이상 스케쥴이 연장 될 수 없음을 나타낸다.

이 두개중 하나의 제약조건이라도 만족되지 않을 경우에는 현재의 스케쥴로는 더이상의 스케쥴 연장이 불가능하게 된다. 따라서 남아 있는 대상재들에 대해서는 더 이상 제약조건을 검사할 필요가 없게 되는 것이다.

4) 타당성 검토

CLP(R)을 이용하여 개발된 프로토타입을 테스트한 결과 다음과 같은 사실을 알 수 있었다. 즉, Look ahead를 이용해서 탐색영역을 줄이는 방법을 사용한 결과, 가장 소모적인 Enumeration과 비교하여 가능해가 없을 경우에는 매우 빠른 시간 내에 스케쥴이 불가능함을 보여주었으며, 가능해가 존재 했을 경우에는 현재의 스케쥴에서 약간의 수정(Backtracking & Reinstantiation)을 통해 또 다른 가능해를 연속적으로 구할 수 있는 장점을 보여 주었다. 이와같은 과정을 프로토타입에서 실제 데이타를 가지고 실행해 본 결과 만족할 만한 탐색영역 축소의 효과가 있음을 발견할 수 있었다. 또한 최종 구현될 시스템의 실행속도가 CLP(R)의 프로토타입보다는 빠를 것이라는 점에서 실행시간의 상한치를 구하는 역할도 하였다. 이와같은 결론은 이 프로토타입을 실제 시스템에 적용해도 좋다는 것으로 받아들일 수 있게 되었다. 또한 이 프로토타입은 C 언어로의 변환이 용이하기 때문에 실제 시스템을 구축하는데 상당한 도움을 줄 수 있었다.

4. 시스템 구성 및 작업단위편성 과정

본 연구에서는 포항제철의 냉연공정의 작업단위를 편성하기 위해서 크게 3가지의 시스템으로 구성된 작업단위편성 시스템을 개발하였다. 먼저 지시 대상재에서 받은 원래 데이터를 필요한 형태로 바꾸어주는 전처리(Preprocessing)시스템, 실제로 작업단위를 편성하는 루틴이 들어있는 작업 단위 편성시스템, 마지막으로 편성된 작업단위의 코일 데이터를 다시 main frame에서 처리할 수 있는 형태의 데이터로 변환시켜주는 후처리(Postprocessing)시스템으로 구성되어 있다. 이와같은 시스템 구성을 그림 2에 나타내었다.

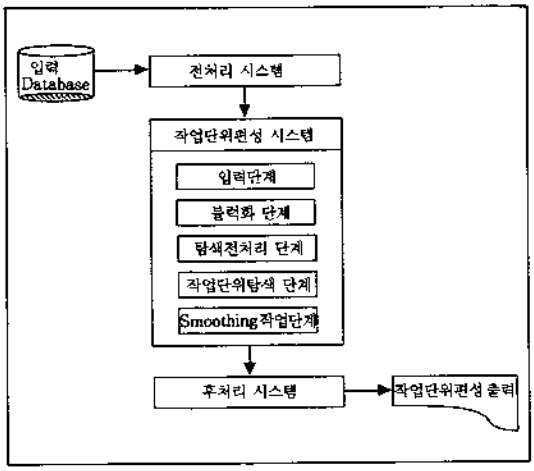


그림 2. 시스템의 구성도

4.1 전처리시스템

이 시스템은 main frame에서 덤프받은 ASC－Ⅱ형태의 데이타 베이스(DB)화일에서 작업단위를 편성하기 위해서 필요한 필드만 선택하는 과정이다.

실제로 main frame에서 덤프받는 화일은 표 1과 같은 형태로 되어있다.

표 1과 같은 형태의 화일에서 작업단위를 편성하기 위해서 필요한 정보인 폭, 열연코일 두께, 중량, X－Ray치, 탄소함유량 등의 데이터를 추출한다. 이 전처리 시스템에 의해서 만들어진 화일은 표 2와 같은 형태로 구성되어 있다. 코일번

표 1. main frame에서 덤프받은 DB화일

L5241736	002301018009560W04C7Y	003000000LCLCF31B
010921130CSP1－HG	CC1HG	00350010020350SMD5CHXX
M008380	002301018014250W04B7NNW	002700010KGSDK23N
10930110JS－SGCC	GCGG	00320010000320SMSR XX
M005330	002301018012690W04B7NNW	0043000000KGLDK23P
020930110JS－SGCC	GCGG	00380010000380SMSR KK
L610740	002300996014140W04A2BQY	003600010KCLCC17R
020921130CESP2－C	DVEC	00350009720352SSD7CSAG
L605280	002300951013900W04A2BQY	003900010KCLCC17S
020921130CESP2－C	CVEC	00350009270352SSD7SCAG
L610650	002300947014330W04A2BQY	003900010KCLCC17Q
011921220CESP2－C	CVEC	00350009240352SSD7CSAG
L610670	002300948014520W04A2BQY	003900010KCLCC17Q
011921220CESP2－C	CVEC	00350009240352SSD7CSAG
M000510	002300947013660B04CPY	004600010KBLDD07F
050930110JS－MRT4CS	BRC4	00270009330274MSB2T4BD
M000620	002300947013660B04CPY	0046000010KBLDD07F
050930110JS－MRT4CA	BRC4	00270009330274MSB2T4BD
M000520	002300947013630B04CPY	004600010KBLDD07F
050930110JS－MRT4CA	BRC4	00270009330274MSB2T4BD

표 2. 전처리 작업을 마친후의 데이터화일

코일번호	폭	두께	X－Ray	탄소 함유량	납기	중량
C01	1018	2.3	0.350	0.027	921130	19.6
C02	1218	2.5	0.380	0.030	930110	12.7
C03	1018	2.1	0.320	0.036	930110	14.2
C04	996	2.5	0.284	0.043	930110	18.8
C05	948	3.9	0.352	0.043	930110	18.0
C06	996	2.3	0.332	0.043	921130	13.9
C07	996	2.3	0.332	0.038	921220	14.5
C08	948	2.3	0.320	0.039	930110	17.6
C09	996	2.0	0.352	0.039	921130	14.1
C10	996	2.1	0.332	0.038	930110	14.3
C11	948	2.7	0.284	0.038	930110	18.2
C12	948	2.5	0.320	0.043	930110	18.1
C13	948	2.2	0.332	0.043	930110	18.1

호는 원래 표 1에서 보듯이 L524173, M008380 등과 같으나, 표 2에서는 편의상 C01, C02,…와 같이 나타내었다.

위와 같은 형태의 화일을 TCM공정의 경우에는 극박, 중박, 후물, 전강의 4가지로 구분하여 얻게되며, 이 전처리 작업에 의해서 작업단위를 편성하기 위한 데이터화일을 얻게 되는 것이다. (본 논문의 예는 후물에 대한 것임).

4.2 작업단위 편성 시스템

본 시스템은 크게 입력단계, 유사한 코일을 묶는 블럭화 단계, 그리고 우선 순위에 따라 블럭내 코일을 나열하여 작업편성을 조정하는 Smoothing작업 단계로 구성된다.

입력단계

전처리 작업을 해서 얻은 표 2와 같은 데이터화일을 폭과 X-Ray치를 기준으로 정렬된(광폭에서 협폭으로 정렬하며, 다시 동일폭 내에서는 X-Ray치의 내림순으로 정렬함) 표 3과 같은 데이터 화일이 입력된다.

표 3. 입력 데이터 화일

코일번호	폭	두께	X-Ray	탄소함유량	납기	중량
C02	1218	2.5	0.380	0.030	930110	12.7
C01	1018	2.3	0.350	0.027	921130	19.6
C03	1018	2.1	0.320	0.036	930110	14.2
C09	996	2.0	0.352	0.039	921130	14.1
C06	996	2.3	0.332	0.043	921130	13.9
C07	996	2.3	0.332	0.038	921220	14.5
C10	996	2.1	0.332	0.038	930110	14.3
C04	996	2.5	0.284	0.043	930110	18.8
C05	948	3.9	0.352	0.043	930110	18.0
C13	948	2.2	0.332	0.043	930110	18.1
C08	948	2.3	0.320	0.039	930110	17.6
C12	948	2.5	0.320	0.043	930110	18.1
C11	948	2.7	0.284	0.038	930110	18.2

블럭화 단계

위의 표 3과 같은 코일 단위로 작업단위를 편성하면 가능해의 탐색시간이 길어지는 문제점이 있다. 이렇게 입력받은 코일 정보를 갖고 작업을 편성하는 것보다는 이 코일들중에서 폭과 X-Ray치가 동일한 코일들을 한개의 블럭으로 만든 다음, 이러한 블럭단위로 작업단위를 찾기 위한 탐색작업을 함으로써 탐색시간을 상당히 줄일 수 있다. 블럭화 작업을 해준 결과의 예는 다음 표 4와 같다. 예를들어 블럭번호 5에는 코일 C06, C07, C10이 합하여진 것으로 블럭의 중량은 그 블럭에 해당하는 코일들의 중량을 합한 것으로 나타낸다. 한 블럭에 대응하는 코일들의 두께 및 납기는 서로 다를 수 있는데 이와같은 정보는 계속적으로 지니고 있으며 Smoothing 작업단계에서 사용된다.

표 4. 코일을 블럭화해서 만든 블럭 화일

블럭번호	폭	X-Ray치	중 량	코 일
1	1218	0.380	12.7	C02
2	1018	0.350	19.6	C01
3	1018	0.320	14.2	C03
4	996	0.352	14.1	C09
5	996	0.332	42.7	C06, C07, C10
6	996	0.284	18.8	C04
7	948	0.352	18.0	C05
8	948	0.332	18.1	C13
9	948	0.320	35.7	C08, C12
10	948	0.284	18.2	C11

탐색 전처리단계

이제 이 블럭을 직접 이용해서 작업단위를 찾아줄 수도 있지만, CSP에 대한 탐색을 돕는 기법으로써 크게 두가지 기법 즉, 단위편성이 잘되는 순서로 정렬시켜놓고 탐색하는 기법과 Fail First Principle[3]에 근거하여 탐색조건을 가능

한 한 만족하지 않는 순서로 정렬하고 탐색하는 기법이 있다. Fail First 기법은 불필요한 탐색을 반복하지 않으므로 탐색시간을 줄일 수 있다. 이러한 두 기법에는 각각 장, 단점이 있는데 해가 있는 경우에는 잘되는 순서로 전처리 작업을 해줌으로써 존재하는 가능해를 빨리 찾을 수 있으며, 작업단위가 존재하지 않을 경우에는 조건을 만족하지 않는 순서로 전처리 작업을 해줌으로써 빠른 시간안에 작업단위가 존재하지 않음을 보여줄 수 있다. 이런 작업이 여기서 말하는 탐색 전처리작업으로써, 이작업을 통해서 다음 표 5와 같이 각 블럭으로 구성된 배열의 인덱스를 갖는 PRE[]라고 하는 배열을 만들 수 있다.

표 5. 잘되는 순서로 전처리 작업을 한후 만들어진 PRE[] 배열

PRE[1]=1 PRE[2]=2 PRE[3]=3 PRE[4]=4
PRE[5]=5 PRE[6]=6 PRE[7]=7 PRE[8]=8
PRE[9]=9 PRE[10]=10

작업단위 탐색단계

실제로 이 탐색단계는 Backtracking으로 블럭 단위의 작업순서를 결정하는 단계로서 다음과 같이 7단계의 과정이 반복된다.

Step 0. 초기화 단계

초기의 Partial Schedule은 빈 것으로 가정하고 첫 대상 블럭을 무조건 Partial Schedule에 추가한다.

(여기에 Partial Schedule이란 탐색과정의 현시점까지 이루어진 작업단위 부분적 편성을 말한다)

Step 1. Done_Check 단계

Partial Schedule에 편성되어 있는 블럭들의 총톤수를 계산하여 원하는 작업단위 편성량보다 많은가를 체크한다. 많으면 'SUCCESS'를 리턴하고 Step 8로 가며, 그렇지 않으면 Step2로 간다.

Step 2. Look_ahead 단계

현재 편성된 Partial Schedule의 총톤수와 Partial Schedule의 제일 마지막에 있는 블럭보다 폭이 작은 블럭들의 총톤수를 더하여 작업단위 편성량보다 많은가를 체크하고, Partial Schedule의 첫블럭과 아직 Partial Schedule에 편성되지 않은 블럭들중에서 폭이 가장 큰 블럭과의 차가 주어진 범위이내에 드는지를 체크한다.

위 조건을 만족하면 Step 3으로 가고, 그렇지 않으면 Step 6으로 진행한다.

Step 3. Get_Block단계

Partial Schedule의 마지막 블럭보다 폭이 작은 블럭을 불러온다.

Step 4. Consistency 체크단계

작업단위 편성상의 모든 제약조건(2절에 설명된)들을 만족하는 지를 체크한다. 만족하면 Step 5로 가고, 그렇지 않으면 현재 불러 온 블럭을 돌려 보내고 Step 7로 간다.

Step. 5 첨가 단계

대상 블럭을 Partial Schedule의 맨 마지막에 편성해 준다. 또한 Partial Schedule의 정보를 갱신해 준다. 그리고 Step 1로 가서 과정을 반복한다.

Step. 6 Pullback 단계

Partial Schedule의 맨 마지막에 편성된 블럭을 제거시키고 Partial Schedule의 정보를 갱신한다. 그리고 Step 2로 가서 과정을 반복한다.

Step. 7 순환 단계

Partial Schedule의 마지막 블럭보다 폭이 작은 블럭이 있는가를 체크한다. 있으면 Step 3으로, 그렇지 않으면 'FAIL'을 리턴하고 Step 8로 간다.

Step. 8 Success_check 단계

리턴값이 'SUCCESS'이면 해(편성된 블럭 작업단위)를 출력한다. 그렇지 않고, 'FAIL'이면 작업단위 편성이 불가능임을 출력 한다.

다음에는 윗 과정에서는 주요 단계에 대하여 부연 설명을 하고자 한다.

1) Look-Ahead 단계 : 이 단계에서는 Partial Schedule의 마지막 블럭을 기준으로 Step 2와 3.2절에서 언급한 2가지의 제약조건을 사용해서 Looking Ahead작업을 하고 있다. 먼저 마지막 블럭의 폭보다 작은 폭을 갖는 모든 블럭들의 총톤수를 구한후 이 총톤수와 현재 Partial Schedule에 편성된 블럭들의 총톤수를 합하여 우리가 원하는 톤수의 작업단위를 편성할 수 있는지 조사한다. 만일 모두 합했는데도 원하는 톤수를 얻지 못한다면 더 진행해 보아야 원하는 단위 편성을 할 수 없으므로 여기서 가능성이 없다는 메시지를 되돌려 주면서 이 루틴을 끝낸다. 만일 Partial Schedule이 비어 있는 상태, 즉 첫 블럭이었다면 무조건 편성 할 수 있다는 메시지를 되돌려 준다. 또다른 조건에 의해서 처음 몇개의 블럭의 폭이 유별나게 크고 나머지 블럭들의 폭이 작은경우 처음 몇개의 블럭을 붙여보아야 작업단위에는 결국 첨가될 수 없는 블럭들이므로 처음부터 작업단위에서 제거시키도록 한다. 위의 두가지 제약 조건을 모두 만족하는 경우에만 편성하여도 좋다는 메시지를 되돌려 줌으로써 탐색시간을 줄인다.

2) Consistency 체크단계 : 이 단계에서는 해당 블럭에 대하여 2절에서 기술한 제약조건을 만족하는지를 체크한다. 모든 제약조건을 만족하는 경우에 OK라는 메시지를 주며, 위의 조건중에서 한가지라도 만족하지 않는 경우는 NO_OK라는 메시지와 함께 끝낸다.

3) 첨가단계 : 이 단계에서는 위의 단계를 모두 만족하는 경우 Partial Schedule의 내용을 갱신시켜준다. 즉 현재 Partial Schedule의 총톤수, Schedule에 포함된 블럭정보 등을 갱신한다.

4) Pullback단계 : Looking Ahead단계에서 더 이상 진행해 보아야 가망성이 없으므로 Partial Schedule의 마지막에 붙였던 블럭을 떼어버리면서 Partial Schedule의 정보들을 첨가단계에서와 마찬가지로 갱신해준다.

이상의 작업에 의해서 만들어진 작업 단위는 다음 표 6과 같다.(최소 총톤수를 180톤으로 하였을 경우). 본 편성에서 블럭 1, 7이 제외되었음을 볼 수 있다.

표 6. 편성된 예비 작업단위

(적정단위 180톤 경우)

블럭번호	폭	X Ray치	중 량	코 일
2	1018	0.350	19.6	C01
3	1018	0.320	14.2	C03
4	996	0.352	14.1	C09
5	996	0.332	42.7	C06, C07, C10
6	996	0.284	18.8	C04
8	948	0.332	18.1	C13
9	948	0.320	35.7	C08, C12
10	948	0.284	18.2	C11
총 계			181.4	

Smoothing 작업단계

앞의 제약조건에서 설명하였듯이 작업단위편

성에 포함된 코일들은 전후 코일간 X-Ray치 및 두께 등의 편차가 될 수 있는대로 작아야 하므로 블럭간의 교체 또는 한 블럭내에서 코일간의 교체가 필요하다. 이러한 작업이 Smoothing 단계에서 이루어진다. 먼저 표 6의 각 코일의 정렬키값을 (폭*100000)+(X-Ray치*100000)+(열연코일 두께*10)으로 정의하여 정렬키값의 내림차순으로 정렬한다. 정렬된 결과는 다음 표 7과 같다. 여기서 블럭 9내의 코일 순서가 바뀐 것을 볼 수 있다.

표 7. 폭 X-Ray, 두께순서로 정렬된 결과

코일번호	블럭번호	폭	X-Ray	두 께	중 량
C01	2	1018	0.350	2.3	19.6
C03	3	1018	0.320	2.1	14.2
C09	4	996	0.352	2.0	14.1
C06	5	996	0.332	2.3	13.9
C07	5	996	0.332	2.3	14.5
C10	5	996	0.332	2.1	14.3
C04	6	996	0.284	2.5	18.8
C13	8	948	0.332	2.2	18.1
C12	9	948	0.320	2.5	18.1
C08	9	948	0.320	2.3	17.6
C11	10	948	0.284	2.7	18.2

1) 폭이 같은 블럭을 대상으로 한 Smoothing 작업(X-Ray치 기준)

폭이 같은 블럭들(이를 폭 그룹이라하자)을 대상으로 그룹화작업을 해준다. 앞에서의 블럭화작업과 차이점은, 앞에서는 코일단위를 이용해서 블럭화작업을 했는데 여기서는 앞에서 만들어진 블럭단위를 이용해서 우리에게 필요한 형태로 다시 그룹화작업을 해주는 것이다. 이 작업에 의해서 만들어진 결과는 표 8과 같다.

여기서는 동일폭 그룹내의 블럭들의 X-Ray

표 8. 폭을 기준으로 한 그룹화 작업

폭그룹	블럭번호	폭	X-Ray	중 량	코 일
1	2	1018	0.350	19.6	C01
1	3	1018	0.320	14.2	C03
2	4	996	0.352	14.1	C09
2	5	996	0.332	42.7	C06, C07, C10
2	6	996	0.284	18.8	C04
3	8	948	0.332	18.1	C13
3	9	948	0.320	35.7	C12, C08
3	10	948	0.284	18.2	C11

치를 기준으로 Smoothing 작업을 하는데, 전 그룹의 마지막 X-Ray치와 다음 그룹의 첫번째 블럭의 X-Ray치의 차이와 전 그룹의 마지막 X-Ray치와 다음 그룹의 마지막 블럭의 X-Ray치의 차이를 비교해서 차이가 적은 블럭이 다음 그룹의 처음으로 오도록 블럭들을 재 정렬을 해주어야 한다. 즉 X-Ray치가 변화되는 편차를 적게 만들어 주기 위한 작업이다. 이작업에 의해서 만들어진 결과를 보면 표 9와 같다. 표 9를 살펴보면 폭그룹 3에 블럭들의 순서가 바뀐 것을 볼 수 있다.

표 9. 동일폭 그룹내 블럭의 Smoothing된 결과

폭그룹	블럭번호	폭	X-Ray	중 량	코 일
1	2	1018	0.350	19.6	C01
1	3	1018	0.320	14.2	C03
2	4	996	0.352	14.1	C09
2	5	996	0.332	42.7	C06, C07, C10
2	6	996	0.284	18.8	C04
3	10	948	0.284	18.2	C11
3	9	948	0.320	35.7	C12, C08
3	8	948	0.332	18.1	C13

2) 블럭내코일의 Smoothing작업(열연코일 두께 기준)

다음으로 폭과 X-Ray치가 같은 한 블럭내의 코일들의 순서를 재조정하기 위해 블럭에 포함된 코일들의 두께 정보를 활용한다. 이해를 돕기위해 표 9의 결과에 따른 블럭별 코일들을 나열하면 표 10과 같다.

표 10. 블럭별 코일 정보

블럭번호	코일번호	폭	X-Ray	두 께	중 량
2	C01	1018	0.350	2.3	19.6
3	C03	1018	0.320	2.1	14.2
4	C09	996	0.352	2.0	14.1
5	C06	996	0.332	2.3	13.9
	C07	996	0.332	2.3	14.5
	C10	996	0.332	2.1	14.3
6	C04	996	0.284	2.5	18.8
10	C11	948	0.284	2.7	18.2
9	C12	948	0.320	2.5	18.1
	C08	948	0.320	2.3	17.6
8	C13	948	0.332	2.2	18.1

그리고 앞의 경우와 마찬가지로 전후 블럭에 대해서 두께편차가 적도록 재 정렬을 시켜준다. 이 작업에 의해서 나온 결과를 살펴보면 표 11과 같다. 블럭 5내의 코일들의 순서가 변경되었음을 볼 수 있다.

3) 추가 Smoothing 작업

위와 같은 과정에 의해서 Smoothing 작업을 해주고 난 후 폭, X-Ray치, 두께가 같은 그룹내에서 다시 탄소함유량, 납기일 순으로, 즉 정렬키값을(탄소함유량*10)+(999999-납기)*0.001)로 정의하여 내림차순으로 정렬하면 표 12와 같은 결과를 얻을 수 있다. 표 12를 살펴보면 표 11의 블럭 5에 있는 코일 C07, C06의 순서가 고탄소에서 저탄소의 순서로 바뀌었음을 알 수 있다.

표 11. 블럭내 코일의 Smoothing작업결과

블럭번호	코일번호	폭	X-Ray	두 께	중 량
2	C01	1018	0.350	2.3	19.6
3	C03	1018	0.320	2.1	14.2
4	C09	996	0.352	2.0	14.1
5	C10	996	0.332	2.1	14.3
	C07	996	0.332	2.3	14.5
	C06	996	0.332	2.3	13.9
6	C04	996	0.284	2.5	18.8
10	C11	948	0.284	2.7	18.2
9	C12	948	0.320	2.5	18.1
	C08	948	0.320	2.3	17.6
8	C13	948	0.332	2.2	18.1

표 12. 편성된 후물 작업단위
(적정단위 180Ton의 경우)

블 럭	코일번호	폭	두께	X-Ray	탄소함유량	납기	중량
2	C01	1018	2.3	0.350	0.027	921130	19.6
3	C03	1018	2.1	0.320	0.036	930110	14.2
4	C09	996	2.0	0.352	0.039	921130	14.1
5	C10	996	2.1	0.332	0.038	930110	14.3
	C06	996	2.3	0.332	0.043	921130	13.9
	C07	996	2.3	0.332	0.038	921220	14.5
6	C04	996	2.5	0.284	0.043	930110	18.8
10	C11	948	2.7	0.284	0.038	930110	18.2
9	C12	948	2.5	0.320	0.043	930110	18.1
	C08	948	2.3	0.320	0.039	930110	17.6
8	C13	948	2.2	0.332	0.043	930110	18.1

위의 편성된 작업단위에 속해있는 코일들의 전후 폭, X-Ray치, 두께등의 제약조건을 살펴보면, 모든 제약조건들을 만족하는 작업단위가 편성되었음을 알 수 있다.

4.3 후처리 시스템

위와 같은 방법으로 TCM공정에 대해서 작업단위를 편성해 주고나면 다시 편성된 작업단위를 main frame에서 이용할 수 있는 형태의 데이터 화일로 바꾸어 주어야 하는데 이작업을 후처리시스템에서 해준다. 실제로 해주는 작업은 우리가 편성한 코일의 순서를 main frame이 알 수 있는 형태의 데이터 화일로 변환시켜 주는 것인데, 여기서는 처음에 덤프받은 DB화일을 재배열된 표 1과 같은 형태의 DB화일의 형태로 돌려주고 있다.

5. 결 론

본 시스템의 모든 모듈은 C언어로 작성되었으며 IBM 486PC에서 구현되었다. 본 시스템을 포항제철소 현장에서 실제 자료를 갖고 테스트한 결과 실무자로부터 만족스럽다는 반응을 얻었다.

또한, 방대한 데이타들을 블럭화 작업이나 전처리 단계를 통하여 효율적으로 재편성하고, Look-Ahead 단계를 통하여 탐색해야하는 코일들을 미리 제거함으로써 탐색시간을 상당히 단축시킬 수 있었다. 실제 테스트에 의하면 1000개 정도의 코일을 대상으로 PC에서 작업단위편성만을 도출하는 시간은 약 30초 정도 소요된다. 그리고, 본 시스템을 통한 효율적인 작업단위 편성방법은 TCM공정뿐만 아니라, 이와 비슷한 공정에서의 작업단위 편성에도 Consistency 체크모듈만 수정하면 충분히 적용시킬 수 있다.

본 연구를 통한 기대효과로는 첫째, 기존의 수작업을 대신함으로써 인력을 유용하게 활용할 수 있으며, 둘째, 보다 효율적인 작업편성방법을 개발하여 이를 전산시스템화 함으로써 지시편성 단위 적중율을 높일 수 있으며, 셋째, 불필요한 Roll 교체를 줄임으로써 생산성을 높이고 원가의 절감과 품질의 향상을 기할 수 있을 것으로 예상된다.

참 고 문 헌

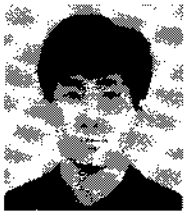
1. 산업과학 기술 연구소, "기본교육교본", 1991.
2. 포항종합제철㈜, 철강일관공정, 87-F-0131, 1987.
3. Haralick, R.M. and Elliot, G.L., "Increasing Tree Search Efficiency for Constraint Satisfaction Problem", *Artificial Intelligence*, Vol. 14, pp.263-313, 1980.
4. Hentenyck, P.V., *Constraint Satisfaction in Logic Programming*, The MIT Press, London, 1989.
5. Haralick, R.M., Davis, L.S., Rosenfeld, A. and Milgram, D.L., "Reduction Operation for Constraint Satisfaction", *Information Science*, Vol 2, No. 5, pp.199-219, 1978.
6. Davis, R.E, "Logic Programming and Prolog Tutorial", *IEEE Software*, Vol 2, No. 5, pp.53-62, September, 1985.
7. Jaffar, J., and Lassea, J.L., "Constraint Logic Programming", *Proceedings of the Conference on Principles of Programming Languages*, Munich, 1987.
8. Lassez, C., "Constraint Logic Programming", *BYTE Magazine*, Special Issue on Logic Programming, August, 1987.
9. Jaffar, J., and Michaylov, S., "Methodology and Implementation of CLP System", *Proceedings of 4th International Conference on Logic Programming*, Melbourne, MIT Press,

pp. 196－218, 1987.

10. Heintze, N.C., Nichaylov, S. and Stuckey, P. J., "CLR(R) and Some Electrical Engineering Problems", *Proceedings of the 4th International Conference on Logic Programming*, Melbourne, MIT Press, pp.675－703, 1987.
11. Lassea, J.L., McAloon, K., and Yap, R., "Constraint Logic Programming in Options Trading", *IEEE Expert*, August, 1987.
12. Ostroff, J.S., "Constraint Logic Programming for Reasoning About Discrete Event Processes", *J. of Logic Programming*, Vol. 11, pp.243－270, 1991.

저 자 소 개

전치혁(全治赫)
1954년 3월 23일생. 1977년 서울대 자원공학과, 1979년 한국과학원 산업공학과(석사), 1986년 미국 Univ. of California, Berkeley 산업공학과(공학박사)졸업. 1987년 포항공과대학 산업공학과 부임, 현재 부교수.

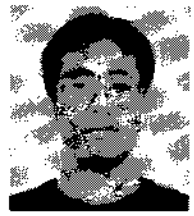


박철순(朴喆淳)
1966년 5월 8일생. 1992년 성균관대 산업공학과 졸업. 현재 포항공과대학 산업공학과 석사과정.

장수영(張秀榮)
1960년 3월 20일생. 1982년 연세대 물리학과, 1983년 미국 Univ. of San Francisco (응용과학 석사), 1988년 미국 Univ. of Michigan 산업공학과(박사) 졸업. 1989년 포항공과대학 산업공학과 부임, 현재 조교수.

강전태
1958년 2월 6일생. 1987년 인하대 금속공학과 졸업. 1987년 포항제철㈜ 입사, 현재 생산관리부 박판공정과 재직.

이승만(李承晩)
1968년 7월 23일생. 1991년 한양대 산업공학과 졸업, 현재 포항공과대학 산업 공학과 석사과정.



강상엽(姜尙燁)
1969년 8월 18일생. 1992년 포항공과대학 산업공학과 졸업. 현재 동 대학원 석사과정.

최인준(崔仁濬)
1958년 4월 11일생. 1981년 서울대 계산통계학과, 1987년 미국 Univ. of Texas at Austin 계산학과(석사), 1991년 동 대학원(경영정보 박사) 졸업. 1991년 포항공과대학 산업공학과 부임, 현재 조교수.