최대 크기 일괄처리 방식에 대한 연구

Maximum batch cover for order consolidation

민윤홍*, 홍성필*, 권순우*, 최병천**, 장수영***

* 서울대학교 산업공학과 (myh@snu.ac.kr, sphong@snu.ac.kr, soonwoo@snu.ac.kr)

** u-컴퓨팅 원천기술 개발지원센터 (cbc@optima.snu.ac.kr)

*** 포항공과대학교 산업경영공학과 (syc@postech.ac.kr)

초 록

본 논문에서는 서로 다른 주문들의 일부를 일괄적으로 처리할 수 있을 때, 최대 개수의 작업 단위로 최대한 많은 수의 주문을 처리하는 문제를 다룬다. 무향그래프와 배치 크기 λ로 표현되는 이 문제의 일반적인 경우는 NP-hard이며, 나무일 때 다항시간 안에 최적해를 찾을 수 있음이 증명되어 있다. 본 논문에서는 이분 그래프의 경우에도 NP-hard임을 보이고, 이 경우에 절대근사 알고리즘과 상대근사 알고리즘을 제시한다.

키워드: 일괄처리, 근사해법

1. 서 론

다양한 주문들을 처리해야 하는 생산 공정에서 서로 다른 주문들을 일괄적으로 처리하는 배치 생산이 가능할 경우 생산 공정의 효율을 높일 수 있다. 주문 집약 문제는 가능한 적은 수의 배치로 전체 주문을 처리하는 최소화 문제와 주어진 주문을 배치로 처리할 때, 배치 최대 크기만큼 가능한 많은 수의 주문을 처리하는 최대화 문제로 나눌 수 있다.

최대화 문제는, 다음과 같이 무향그래프를 이용하여 표현할 수 있다. 주어진 주문들과 주문량을 마디와 각 마디의 가중치로 표현하고, 서로 다른 두 주문이 서로 배치 처리가 가능할 경우 해당하는 두 마디를 호로 연결한다. 이런 그래프에서 배치크기 λ로, 각 마디의 주문량을 넘지 않는 최대 개수의 배치를 찾는 것이 최대화 문제이다. 최대화 문제의 경우, NP-hard이며, Max-SNP-hard 임이 증명되어 있다[1]. 또한, 그래프가 나무 구조를 가지는 경우 다항시간 알고리즘이 존재한다[1]. 하지만, 다른 구조의 그래프에서의 문제의 계산 복잡도에 대해서는 알려진 바가 없으며, 특수한 그래프와 일반적인 그래프 모두에서의 근사 가능성에 대해서도 알려진 바가 없다.

본 논문에서는 이분그래프에서의 최대 개수로 주어진 주문을 일괄 처리 하는 문제를 다룬다. 2장에서는 이분 그래프에서 우리 문제가 NP-hard에 속함을 보이고, 3장에서는 상대 근사 알고리즘과 절대 근사 알고리즘. 그리고 4장에서는 결론을 제시할 것이다.

2. NP-completeness 증명

NP-complete 문제인 PARTITION 문제[2]를 이용하여 이분 그래프에서의 최대크기 주문집약 문제가 NP-complete임을 증명할 것이다. PARTITION는 다음과 같이 정의된다.

PARTITION

Instance : 유한 집합 $A=\{1,...,n\}$, 양의 정수 a_i, $i=1,...,n$

Question : 다음을 만족하는 A의 부분집합 A′가 존재하는가?

$$\sum_{i\in A'} a_i = \sum_{a\in A\setminus A'} a_i$$

일반적인 주문집약 문제의 결정 문제는 다음과 같다.

MAX-ORDER-CONSOLIDATION(MOC)

Instance : 무향 그래프 G=(V,E), 양의 정수 $w(v)$, $v\in V$, 배치크기 B, 양의 정수 K

Question : 배치 개수가 K인, 각 마디의 가중치를 넘지 않는 배치크기 B의 주문집약이 존재하는가?

Theorem 2.1 이분그래프에서 MOC는 NP-complete이다.

Proof MOC는 NP 임을 쉽게 알 수 있다.

PARTITION의 각 인스턴스 X를 a_i를 각 호의 가중치로 갖고, 배치 크기(B)가 $\frac{1}{2}\sum_{i=1}^{n} a_i$이고, $K=n$인 그림 1과 같은 $(2,n)$-완전이분그래프 상에서의 MOC로 변환할 수 있다. 명확하게 이 변환은 다항시간 안에 할 수 있음을 알 수 있다.

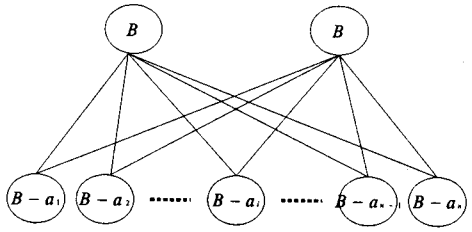

그림 1. $(2,n)$-완전이분그래프

변환된 MOC의 인스턴스를 Y라고 하자. X가 yes-instance이고 이 때의 집합 $\overline{A}$를 고려하자. $\overline{A}$에 속한 원소와 대응되는 가중치 값을 갖는 호들을 root 1과 배치를 만들고, 나머지 호들을 root 2와 배치를 만들면, 총 n개의 배치가 완성이 된다.

Y의 가능한 배치 크기가 n개 이상이라고 하자. 위 그래프 상에서 총 마디의 가중치 합은 nB이므로 만들 수 있는 배치의 최대치는 n개 이다. 이는 배치를 완성했을 때 버려지는 가중치가 0이 되어야 하고, 2개의 root 호가 혼자 배치를 이루면 안 된다는 것을 의미한다 (이럴 경우 만들 수 있는 배치의 개수는 항상 n개보다 작다). 그러므로 Y의 가능해는 2개의 root 호의 모든 가중치를 나머지 호들이 배치를 이루도록 사용되는 특성을 갖는다. 해는 root 1과 배치를 이루는 호들의 집합(A_1)과 root 2와 배치를 이루는 호들의 집합(A_2)으로 나뉘면 $\sum_{i\in A_1} a_i = \sum_{i\in A_2} a_i = B$임 만족함을 알 수 있다. 즉, A_1는 PARTITION의 해가 된다.

$(2,n)$-완전이분그래프 상에서의 MOC는 이분그래프에서의 MOC의 특수한 경우이므로, 이분 그래프에서의 MOC는 NP-complete 이다. ■

3. 상대근사 및 절대근사 알고리즘

2장에서 NP-complete 증명에 사용한 (2,n)-완전이분그래프의 경우에 2-factor 상대근사알고리즘이 존재한다. 다음의 알고리즘을 살펴보자.

알고리즘 1.
step 1. Root A를 중심으로 하는 별그래프와 Root B를 중심으로 하는 별그래프에서 MOC를 푼다.
step 2. 두 결과 중 큰 쪽을 적용하여 배치 방식을 결정한다. 이 때의 결과를 ALG1이라 하자.
step 3. step 2의 결과를 적용한 이후에 남은 가중치를 이용하여 다른 한쪽의 별그래프에서 다시 MOC를 푼다. 이 때의 결과를 ALG2라 하자.
step 4. ALG1+ALG2를 결과로 리턴한다.

알고리즘 1에서 MOC를 푸는 서버루틴은 [1]에서 언급한 STAR-ALGORITHM을 적용한다.

Theorem 3.1 알고리즘 1의 결과(ALG)는 최적해의 1/2배보다는 크다.
Proof 원래 문제의 최적해(OPT)를 이용하여 가중치로 유도된 원래 그래프 G의 서브그래프 G′를 만들 수 있다. G′는 최적해를 이용하여 두 개의 별그래프 G1과 G2로 분리할 수 있다. 이 중 가중치의 합이 더 큰 별그래프를 G1이라 하자. 알고리즘 1의 step 2의 결과는 G1의 결과보다 같거나 크다. 왜냐하면 더 큰 가중치를 가지는 같은 그래프에서 STAR-ALGORITHM을 적용한 것이기 때문이다. 또한, 알고리즘 1의 결과는 ALG1의 2배보다는 작다. 따라서,

$$ALG1 \geq OPT(G1)$$
$$2ALG1 \geq OPT(G1) + OPT(G2) = OPT(G)$$

즉,

$$ALG1 + ALG2 \geq ALG1 \geq \frac{1}{2} OPT(G1)$$

■

알고리즘 1은 (2,n)-완전이분그래프보다 일반적인 (2,n)-이분그래프에서도 동일한 분석이 가능하다.

Corollary 3.2 일반적인 (n_1,n_2)-이분그래프에 알고리즘 1을 반복적으로 적용하면 $\frac{n}{2}$-factor 상대근사 알고리즘을 얻을 수 있다.
Proof Theorem 3.1의 증명과 같은 분석으로 다음의 관계를 얻을 수 있다.

$$ALG \geq ALG1 \geq \frac{1}{n_1} OPT(G)$$

n_1과 n_2중 크기가 작은 쪽을 n_1이라 하면, $n_1 \leq \frac{n}{2}$를 만족하므로, $\frac{1}{n_1} \geq \frac{2}{n}$이다. 따라서,

$$ALG \geq \frac{2}{n} OPT(G)$$ ■

(2,n)-완전이분그래프의 경우 다음의 알고리즘 2를 적용할 경우 절대 근사가 가능하다.

알고리즘 2.
step 1. Root 마디 A와 root 마디 B를 합친 가상의 마디 C를 만들어, 원래의 그래프를 별구조로 만든 다음 STAR-ALGORITHM을 적용한다.
step 2. step 1의 결과에서 A와 B에 걸쳐서 만들어진 배치를 제거한다.

step 2의 과정은 그림 2와 같이 표현할 수 있다.

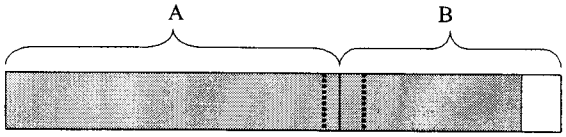

그림 2. 알고리즘 2의 step 2

즉, 점선으로 표현된 배치의 경우 불가능한 배치이므로 이러한 배치를 제거한다.

Theorem 3.3 알고리즘 2의 결과(ALG)는 최적해(OPT)와의 차이가 1 이하이다.
Proof step 1의 결과는 원래 문제의 최적해의 상한을 제공한다. 따라서 step 1 적용 후에 남은 가중치의 합(L)은 최적해에 의해 남은 가중치의 합(L*)의 하한이 된다. 즉, $L \le \sum_{i=1}^{n} w(v_i) - \lambda OPT = L^*$이다. Step 2에서 기껏해야 하나의 배치만이 제거됨을 그림 2에서 확인할 수 있다. Step 2를 적용한 이후에 남은 가중치의 합은

$L + \lambda = \sum_{i=1}^{n} w(v_i) - \lambda ALG$를 만족한다. 따라서,

$$\sum_{i=1}^{n} w(v_i) - \lambda(ALG) \le \sum_{i=1}^{n} w(v_i) - \lambda(OPT-1)$$

$$\lambda ALG \ge \lambda(OPT-1)$$

$$ALG \ge OPT-1$$

■

그림 3과 같은 (2,n)-이분그래프를 생각해보자.

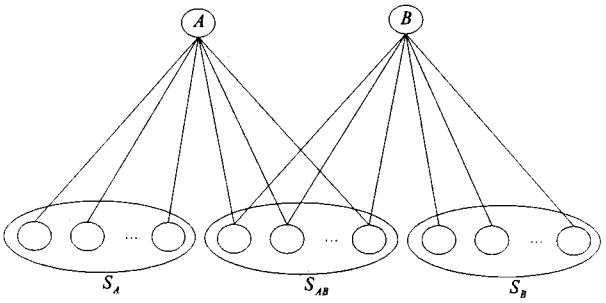

그림 3. (2,n)-이분그래프

각 leaf 마디에 $\alpha_i = w_i - \left\lfloor \frac{w_i}{\lambda} \right\rfloor$,

$\beta_i = \lambda - \alpha_i$를 정의한다. Root 마디 A와 연결되어 있는 마디의 집합을 S_A, root 마디 B와 연결되어 있는 마디의 집합을 S_B, A와 B 모두와 연결되어 있는 마디의 집합을 S_{AB}라고 하자. 다음과 같이 호를 제거하여 보자. S_A와 S_B에 속한 마디들에 대해서 각 집합에 속한 마디를 β_i가 작은 것부터 나열하여 그 값을 합할 때, A와 B의 가중치를 넘게 되는 마디에 연결된 호를 제거한다. 그리고 이 마디들을 각각 S_A와 S_B에서 제거한다. 그러면 남은 마디들은 다음을 만족하게 된다.

$$\sum_{i=1}^{|S_A|} \beta^{A_i} \le w_A, \sum_{i=1}^{|S_B|} \beta^{B_i} \le w_B$$

S_A에서 root 마디와 함께 배치를 만드는 마디의 β_i의 합이 W_A보다 클 수 없다. 또한, 최적해에서 β_i가 순차적으로 작은 것부터 존재하지 않는다면, 즉 $|S_A| = k$라고 할 때 어떤 $\beta_i \le \beta_j$가 존재해서 j는 S_A에 속하고 i는 속하지 않는다면, i와 j를 교환해서 같은 수의 배치 크기를 유지하면서 남은 가중치를 더 크게 만들 수 있게 된다. 따라서 일반성을 잃지 않고 위의 조건을 만족하는 (2,n)-이분그래프에 대해서만 고려한다. 알고리즘 2에 기반한 다음의 알고리즘 3를 적용해 보자.

알고리즘 3.
step 1. Root 마디 A와 root 마디 B를 합친 가상의 마디 C를 만들어, 원래의 그래프를 별구조로 만든 다음 STAR-ALGORITHM을 적용한다.
step 2. step 1의 결과에서 S_A에 속하는 마디들을 모두 A쪽에 배정하고, S_B에 속하는 마디들을 모두 B쪽에 배정한다.
step 3. S_{AB}에 속한 마디들 중에서 배치가 불가능한 배치를 제거한다.

step 2와 3의 과정은 그림 3와 같이 표현할 수 있다.

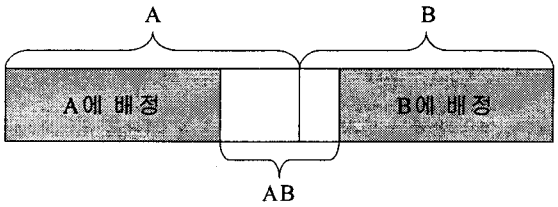

그림 3. 알고리즘 3의 step 2

이 때 알고리즘 2와 같이 AB 구간 사이에서 최대 한 개의 마디만이 배치를 이루지 못하고 제거된다.

Theorem 3.4 알고리즘 3의 결과(ALG)는 최적해(OPT)와의 차이가 1 이하이다.
Proof theorem 3.3과 같은 방법으로 증명이 가능하다.■

Corollary 3.5 일반적인 (n_1, n_2)-이분그래프에 알고리즘 3을 반복적으로 적용하면 $\left\lfloor \frac{n}{2} \right\rfloor - 1$ 절대 근사알고리즘을 얻을 수 있다.

4. 결 론

최대크기 주문집약 문제는 NP-hard에 속할 뿐만 아니라 Max-SNP-hard에 속하는 문제로 PTAS가 존재하지 않는 어려운 문제이다. 만약 이 문제가 Max-SNP-complete에 속한다면, 상수 상대 근사알고리즘이 존재하지만, Max-SNP-complete인지에 대해서 밝혀진 바가 없으며, 알려진 상수 상대 근사알고리즘도 없다. 본 논문에서는 이 문제의 다항 알고리즘이 존재하는 경우인 나무, 특히 별그래프의 경우의 단순한 확장인 (2,n)-완전이분그래프에서도 이 문제가 NP-hard임을 보였고, (2,n)-이분그래프에서의 상수 근사알고리즘을 보였다. 이 알고

리즘은 일반적인 이분그래프에도 적용이 가능하지만, 본 논문에서는 입력값의 다항함수의 근사알고리즘으로의 분석만을 보였다.

이분 그래프에 대한 추후 연구 방향으로는 일반적인 이분 그래프에서 상수 근사 알고리즘이 존재하는지, 혹은 이분 그래프에서의 근사 불가능성에 대한 연구가 가능하다. 또한 보다 일반적인 그래프에서의 주문집약 문제에 대한 근사해법 개발에 대한 연구가 필요하다.

참 고 문 헌

[1] HARK-CHIN HWANG and SOO Y. CHANG, "Order consolidation for batch processing", Journal of Combinatorial Optimization, Vol. 9(2005), pp. 121-138

[2] M.R. Garey and D.S. Johnson, Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman: San Francisco, CA, 1979.