최적화근사해법

홍 성 필

서울대학교 산업공학과
최적화 연구실

A supplementary lecture note for the book *Approximation algorithms* by V. Vazirani

2007년 봄학기

Part I

Chapter 1, 2

NP-optimization problems

정의 1.1

An NP-optimization, Π consists of the followings.

- A set of *valid instances*, D_Π, recognizable in polynomial time. All the numbers are rational. The size of an instance $I \in D_\Pi$, denoted by $|I|$ is the number of bits needed to write I in binary.
- Each instance $I \in D_\Pi$ has a nonempty set of *feasible solutions*, $S_\Pi(I)$. Every solution $s \in S_\Pi(I)$ is of length polynomially bounded in $|I|$. Also, there is polynomial time algorithm that, given I and s, decides whether $s \in S_\Pi(I)$.
- There is a polynomial time computable *objective function* $\mathrm{obj}_\Pi : S_\Pi(I) \to \mathbb{Q}_+$.
- Π is specified to be either a *maximization problem* or a *minimization problem*.

When Π is a minimization problem, $s \in S_\Pi(I)$ is an optimal solution for an instance I if $\mathrm{obj}_\Pi(s)$ is the largest among the feasible solutions of $S_\Pi(I)$.

$\mathrm{OPT}_\Pi(I)$ or simply OPT will denote the largest objective value.

Approximation algorithms

Let Π be a minimization (maximization) problem, and $\delta : \mathbb{Z}_+ \to \mathbb{Q}_+$ a function with $\delta \geq 1$ ($\delta \leq 1$, resp.). An algorithm $\mathcal{A}$ is said to be *factor* δ *approximation* algorithm, or δ-*approximation* of Π if, on each instance I, $\mathcal{A}$ produces a feasible solution $s \in S_\Pi(I)$ such that

$$\mathrm{obj}_\Pi(s) \leq \delta(|I|)\mathrm{OPT}_\Pi(I)$$

$$(\mathrm{obj}_\Pi(s) \geq \delta(|I|)\mathrm{OPT}_\Pi(I), \mathrm{resp.}).$$

정의 1.2

(**Vertex Cover**) Given an undirected graph $G = (V, E)$, and a cost function $c : V \to \mathbb{Q}_+$, find a minimum cost *vertex cover*, namely $U \subseteq V$ such that every edge has at least one endpoint in U.

Notation: $c(v) = c_v$ $c(U) = \sum_{v \in U} c_v$.
c_v가 모두 1일 때, Cardinality Vertex Cover (CVC) 라고 부른다.

Lower bounding OPT

NP-hard NP-최적화 문제의 OPT를 구하는 것은 해를 구하는 것과 마찬가지로 NP-hard이다 (왜?).
따라서 근사알고리듬을 디자인 할 때는 OPT를 대신할 수 있는 값이 필요하다. 다항시간 계산가능한 OPT의 하한(lower bound) (최소화 문제의 경우)을 보통 사용한다.
이러한 Lower bounding을 CVC 예를 통해 살펴 보자.

정의 1.3

(**Matching**) Given an undirected graph $G = (V, E)$, an edge set $M \subset E$ is called a matching if no two edges of M shares an endpoint.

성질 1.4

$|M| \leq |C|$ $\forall$ matching M and cover C.

알고리듬 1.5

Find a maximal matching M of G and return the M-covered vertices.

$s \in S$ is *maximal* if no other element of S is "larger than" s and *maximum* if s is "larger than" or equal to any other element.

정리 1.6

Algorithm 1.5 is a 2-approximation of CVC.

Proof

Some questions

1. A better analysis of Algorithm 1.5 is possible?
2. A better algorithm with the same lower bounding scheme?
3. A better approximation?

Well-characterized problems Π

정의 1.7

결정문제 $\Pi \in \mathrm{NP} \iff \Pi$ has a YES-certificate: Π의 답이 '예'일 때, 이를 다항시간에 확인하게 하는 근거가 존재.

성질 1.8

$\mathrm{P} \subseteq \mathrm{NP}$.

정의 1.9

$\Pi \in \mathrm{NP} \iff \Pi^c \in \mathrm{co-NP}$.

성질 1.10

$\mathrm{P} \subseteq \mathrm{co-NP}$.

VC: G는 $|C| \le k$인 vertex cover C를 가지는가?
VC^c: G의 모든 vertex cover C는 $|C| > k$인가?

• VC가 NP에 속하는 것은 쉽게 알 수 있다. YES-certificate은 쉽게 만들 수 있기 때문이다. 즉, VC^{c}는 co-NP에 속한다. 그러나, VC는 YES-certificate과는 달리, 쉬운 NO-certificate (VC^{c}의 YES-certificate)이 존재하지 않으며, 실제로 존재하지 않는다고 받아들이고 있다.

추측 1.11

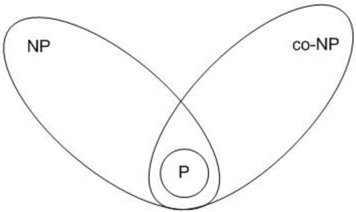
$\mathrm{VC}^c \notin \mathrm{NP}$ *(equiv,* $\mathrm{VC} \notin \mathrm{co-NP}$*)*.

그러면 , 다음과 같은 추측이 역시 성립하게 된다.

추측 1.12

$\mathrm{NP} \neq \mathrm{co-NP}$.

Widely accepted picture



성질 1.13

추측 1.12을 가정하면 어떠한 NP*-hard 문제도 No-certificate을 가질 수 없다.*

(왜인지 설명할 것.)

정의 1.14

결정문제 Π well-characterized. $\Leftrightarrow \Pi \in \mathrm{NP} \cap \mathrm{co-NP}$.

앞의 논의에 의하여 Π가 well characterized라는 것은 다항알고리듬을 가진다는 강한 '방증'이 된다.
역사적으로, 이분그래프에서의 마디커버, 최대 짝짓기 (matching) 문제, 그리고 선형계획(LP)은 well-characterization을 갖는다는 사실에 근거하여 다항알고리듬을 추구했던 대표적인 예.

예 1.15

무향 그래프 G가 이분그래프이면 최대 짝짓기와 최소 vertex cover의 크기는 같다. (König의 정리)
따라서 VC는 이분 그래프에서는 well-characterized. 실제로 이분그래프에서 vertex cover문제는 다항알고리듬이 존재한다.

참고 1.16

이분그래프의 가정이 없으면 두 값이 같지 않은 예를 쉽게 만들 수 있다. 예를 들어, 홀수 회로. 또한 피터슨 그래프는 완전짝짓기를 갖으면서도 최소 VC의 크기가 완전짝짓기의 크기 5 보다 크다. 이는 피터슨 그래프가 마디를 공유하지 않는 두개의 길이 5인 홀수회로를 갖기 때문인데, 각각 최소 3개의 마디가 필요하기 때문이다.

예 1.17

홀수 마디커버와 최대크기 짝짓기 (Tutte-Berge Formula)

예 1.18

LP-duality

따라서, 이분그래프의 짝짓기와 같이 최대크기 짝짓기, 그리고 LP는 well-characterized.
많은 min-max 관계는 LP-duality로 해석할 수 있다. (König의 정리와 Tutte-Berge Formula도 마찬가지.)
뒤에 보겠지만 LP-duality는 다항알고리듬 뿐만 아니라 근사해법 개발에도 매우 유용하다.

집합커버(Set Cover)

문제 2.1

Set Cover (SC)
입력: 유한집합 U, $|U| = n$. U의 부분집합의 콜렉션 $\mathcal{S} = \{S_1, \ldots, S_m\}$, $c : \mathcal{S} \longrightarrow \mathbb{Q}_+$.
최적해: 합하면 U가 되는 $\mathcal{S}$의 최소비용 부분 콜렉션.

- 현재까지의 SC 알고리듬들은 $O(\log n)$이나 f의 근사성을 갖는다. (여기서 f는 U의 한개의 원소가 $\mathcal{S}$의 집합에 나타나는 최대 횟수. VC의 경우, $f = 2$.)

현재까지 커버된 마디들을 C라고 할 때, $\mathcal{S}$의 집합 S의 *평균비용*을 다음과 같이 정의:

$$\frac{c(S)}{|S - C|}.$$

Greedy 알고리듬

1. $C \longleftarrow \emptyset$;

2. **while** $C \neq U$ **do**
현재 가장 작은 평균비용(α라고 하자)을 갖는 S의 집합, say, S를 선택; $S - C$의 각 원소 v들의 *가격*, $p(v) = \alpha$로 정의;

$C \longleftarrow C \cup S$;

3. 선택된 집합들을 출력.

기본정리 2.2

U의 원소들을 커버된 순서대로 $v_1, v_2, \ldots, v_n$ 이라고 하자. 그러면,

$$p(v_k) \leq \frac{\mathrm{OPT}}{n-k+1}.$$

증명: 임의의 반복단계에서도, 최적해에서 사용되지 않은 집합을 다 모으면 OPT를 넘지 않는 비용으로 아직 커버되지 않은 마디들의 집합 $U \setminus C$를 모두 커버할 수 있다. 따라서 이 집합 중에는 평균비용이 $\frac{\mathrm{OPT}}{|U \setminus C|}$를 넘지 않는 것이 있다: 최적해 중 남은 집합들을 $S_{i_1}, S_{i_2}, \ldots, S_{i_l}$이라 하면, $\frac{\mathrm{OPT}}{|U \setminus C|} \geq \frac{(c(S_{i_1})+\cdots+c(S_{i_l}))}{|S_{i_1} \setminus C|+\cdots+|S_{i_l} \setminus C|} \geq \min_k \left\{ \frac{c(S_{i_k})}{|S_{i_k} \setminus C|} \right\}$. (Use: 모든 수 양수일 때, $\frac{a_1}{a_2} \leq \frac{b_1}{b_2}, \frac{c_1}{c_2}$이면 $\frac{a_1}{a_2} \leq \frac{a_1+b_1+c_1}{a_2+b_2+c_2}$.)

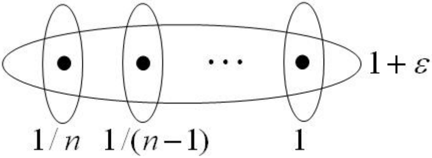
v_k가 커버되는 반복단계를 생각하자. 그러면 $|U - C| \geq n - k + 1$이며, v_k는 알고리듬에 따라 평균비용이 가장 작은 집합으로 커버되기 때문에

$$p(v_k) \leq \frac{\mathrm{OPT}}{|U - C|} \leq \frac{\mathrm{OPT}}{n - k + 1}. \square$$

정리 2.3

위의 알고리듬은 $H_n \equiv 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} \asymp \ln n$의 근사계수를 갖는다.

Tight example



$U = \{1, 2, \ldots, n\}$, $c(\{1, 2, \ldots, n\}) = 1 + \epsilon$,
$c(\{1\}) = \frac{1}{n}$, $c(\{2\}) = \frac{1}{n-1}$, . . ., $c(\{n\}) = 1$.

Essentially the best possible approximation!

Layering

마디 가중치 $w : V \to \mathbb{Q}_+$가 어떤 상수 $c > 0$가 존재하여 $w(v) = c\deg(v)$를 만족하면 *degree-weighted*라고 부르자.

기본정리 2.4

w가 degree-weighted이면 $w(V) \le 2\mathrm{OPT}$.

증명: U를 최적커버라고 하자. 커버이기 때문에 $\sum_{v \in U} \deg(v) \ge |E|$. 따라서 $w(U) \ge c|E|$. 그런데 $w(V) = c\sum_{v \in V} \deg(v) = 2c|E|$. 따라서, $w(V) \le 2\mathrm{OPT}$. □

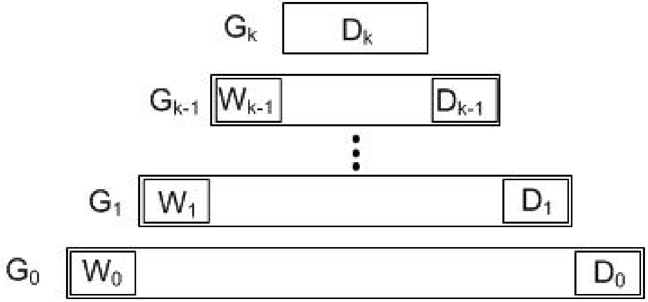
임의의 가중치 w에 대해서, 다음과 같이 *최대 degree-weighted* 가중치를 정의하자: 차수가 0인 마디를 모두 제거한다. $c = \min\{w(v)/\deg(v)\}$를 계산하고 새로운 가중치 $t(v) = c\deg(v)$를 정의한다. 그리고 잔여가중치 $w'(v) = w(v) - t(v)$를 계산한다.

다음과 같은 알고리듬을 적용한다.

$G_0 = G$로 놓는다. G_0로 부터 차수가 0인 마디 집합 D_0를 모두 제거하고 w에 대하여 최대 degree-weighted 가중치를 계산한다. 이 때 0의 잔여가중치를 갖는 마디집합을 W_0라고 하자.

G_1을 마디 집합 $V - (W_0 \cup D_0)$로 유도된 그래프라고 하자. 위의 과정을 잔여가중치를 가중치로 사용하는 G_1에 반복한다.

이 과정은 모든 마디의 차수가 0이 되는 G_k가 생성될 때 까지 반복한다. $C = W_0 \cup \cdots \cup W_{k-1}$을 해로 생성한다.



$t_0, t_1, \ldots, t_{k-1}$을 $G_0, G_1, \ldots, t_{k-1}$의 최대 degree-weighted 가중치라고 하자.

정리 2.5

위의 알고리듬은 임의의 가중치에 대하여 2-근사해법이다.

증명: 우선 C가 마디커버임을 보이자. 만약 아니라면, $V - C = D_0 \cup \cdots \cup D_k$라는 사실로 부터, 어떤 i와 j에 대하여, $u \in D_i$, $v \in D_j$라는 의미이다. $i \leq j$라고 하자. 그러면 (u, v)가 G_i의 호이어야 하며 이는 u가 G_i에서 차수가 0이라는 것에 모순이다.

근사치를 증명하기 위해 C^*를 최적 마디커버라고 하자. 만약 $v \in C \cap W_j$이면 $w(v) = \sum_{i \leq j} t_i(v)$가 된다. 따라서 $w(C) = \sum_{i=0}^{k-1} t_i(C \cap V(G_i))$. 한편 $v \in D_j$이면 $w(v) \geq \sum_{i<j} t_i(v) = \sum_{i \leq j} t_i(v)$ $(t_j(v) = c\deg_{G_j}(v) = 0)$. 따라서 임의의 마디 집합 S에 대하여 $w(S) \geq \sum_{i=0}^{k-1} t_i(S \cap V(G_i))$.

우선 각 i에 대해, $C^* \cap V(G_i)$는 G_i의 마디커버가 된다. 따라서 앞의 기본정리에 의하여
$t_i(C \cap V(G_i)) \leq t_i(V(G_i)) \leq 2t_i(C^* \cap V(G_i))$.

$w(C) = \sum_{i=0}^{k-1} t_i(C \cap V(G_i)) \leq 2 \sum_{i=0}^{k-1} t_i(C^* \cap V(G_i)) \leq 2w(C^*)$.
□

Tight example

$K_{n,n}$ with unit vertex weights.

Part II

Chapter 3

Multicut과 정수다품목흐름문제: 나무의 경우

정의 3.1

메트릭 스타이너나무(MStT) G가 완전호 (complete) 그래프이고 호의 비용이 다음과 같이 삼각부등식을 만족하는 스타이너나무 문제를 *메트릭 스타이너 나무문제*라고 하자:

$$c_{ij} + c_{jk} \geq c_{ik}, \ \forall\, i, j, k \in V.$$

정리 3.2

임의의 스타이너나무 문제를, 근사계수가 유지되도록 하면서, 메트릭 스타이너 나무 문제로 다항변환활 수 있다.

증명: 위의 정의와 같이 스타이너나무 문제가 있을 때, 같은 마디집합을 가진 완전호 그래프 G'에서 스타이너나무 문제를 정의한다. S는 동일하게 놓는다. 비용을 다음과 같이 정의한다.

$$c'_{ij} = G\text{의 } i - j \text{ 최단경로 비용.}$$

호비용 c'이 삼각부등식을 만족하는 것은 당연하다. 또한 G'의 호비용은 G의 호비용보다 크지 않다. 따라서, G'의 최적해는 G의 최적해보다 비용이 크지 않다. G'의 최적해의 호에 대응하는 최단경로의 호를 모두 포함하는 G의 부분그래프는 그 비용이 같다. 만약 회로를 포함하면 회로가 생기지 않을 때까지 호를 제거하여 G의 최적 가능해를 얻을 수 있다. □

MST를 사용한 StT 알고리듬

지금부터 G가 삼각부등식을 만족한다고 가정하자.

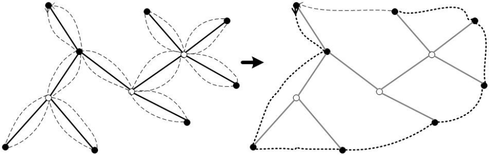
MST 기반 StT 알고리듬

S로 유도된 부분그래프에서 최소신장나무(MST)를 구한다.

정리 3.3

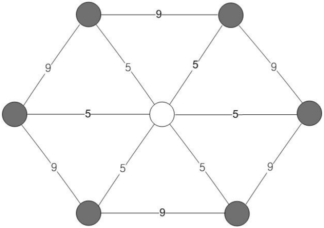
위 알고리듬 해의 비용은 최적 비용의 2배를 넘지 않는다.

증명: 비용 OPT를 가진 최적스타이너 나무를 생각하자. 그리고 호를 두 배로 복사한다. 이렇게 얻은 그래프의 오일러 회로를 구하자. (예를 들어 다음과 같이 깊이우선탐색으로 구할 수 있다.) 그리고 'Short-cutting'을 사용하여, S의 마디들의 해밀턴 경로를 구한다.



삼각부등식에 의하여 해밀턴 경로의 길이는 OPT의 두배를 넘지 않는다. 해밀턴 경로는 S의 마디를 포함하는 걸침나무에 하나이다. 따라서 정리의 증명이 끝난다. □

tight example



메트릭 TSP

정의 3.4

TSP
입력: 완전호 그래프 G, 비음 호 비용 c.
출력: 최소비용 투어, 즉, 모든 마디를 정확히 한번 포함하는 회로.

호비용이 삼각부등식을 만족하면 *메트릭 TSP*라고 하자.

정리 3.5

어떤 다항시간 계산가능 함수 $\alpha(n)$에 대해서도 TSP의 $\alpha(n)$-근사는 불가능하다.

증명: 다음과 같이 그래프 $G = (V, E)$ 위의 HC로 부터 마디집합 V의 완전호 그래프 G'에 정의된 *TSP*로의 다항변환을 생각해보자. G'의 호 e의 비용 c_e는,

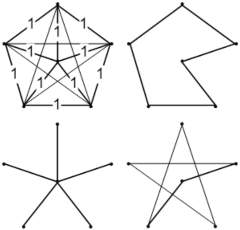
$$c_e = \begin{cases} 1, & e \in E \\ n\alpha(n), & e \notin E. \end{cases}$$

만약 *TSP*의 $\alpha(n)$-근사가 가능하다면, 이를 사용하여 다항시간에 결정문제 HC를 풀 수 있음을 알 수 있다. □

메트릭 TSP를 위한 2-근사

1. G의 MST를 구한다.
2. MST의 호들을 이중복사한다.
3. 오일러 회로를 구한다.
4. 오일러 회로를 경유하며, 마디들이 처음 나오는 순서대로 'short-cutting'을 통하여 투어를 만든다.

Tight example



개선된 $\frac{3}{2}$-근사

1. G의 MST를 구한다.
2. MST의 홀수 차수 마디들로 유도된 G의 부분그래프에서 최소비용 완전짝짓기 M을 구하여, 이 호들을 MST에 추가한다.
3. 위 그래프는 모든 마디 차수가 짝수이므로 존재하는 오일러 회로를 구한다.
4. 오일러 회로를 경유하며 마디들이 처음 나오는 순서대로 'short-cutting'을 사용하여 투어를 만든다.

기본정리 3.6

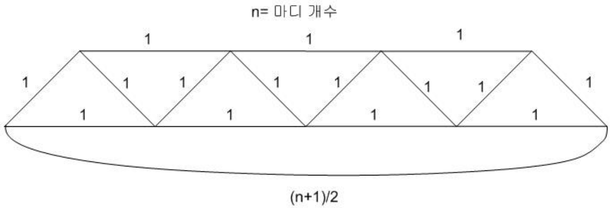
$U \subseteq V$, $|U|$는 짝수, 그리고 M이 U로 유도된 부분그래프의 최소비용 완전짝짓기라고 하자. 그러면 $c(M) \leq \frac{1}{2}\mathrm{OPT}$.

증명: 최적투어 τ를 생각하자. 그리고 τ'을 투어 τ를 경유하며 U에 short-cutting을 적용하며 만든 투어라고 하자. 그러면 $c(\tau') \leq c(\tau)$. 그런데, τ는 U의 두개의 완전짝짓기의 합집합이 된다. 따라서 이중, 비용이 작은 것은 $\frac{1}{2}\mathrm{OPT}$ 를 넘지 못한다. 따라서 최소비용 완전짝짓기도 그러하다. □

위에서 구한 오일러 경로의 비용은

$$\leq c(MST) + c(M) \leq \mathrm{OPT} + \frac{1}{2}\mathrm{OPT} = \frac{3}{2}\mathrm{OPT}.$$

tight example



추측 3.7

메트릭 TSP의 $\frac{4}{3}$-근사가 가능하다.

Part III

Chapter 4

k-터미널절단면, k-절단면문제

정의 4.1

절단면(cut) 연결된 무향그래프 $G = (V, E)$의 V를 분할(partition)하는 임의의 두 집합, $U \cup (V \setminus U)$에 걸쳐있는 호의 집합을 *절단면*(cut)이라고 한다. 만약 $s \in U$, $t \in V \setminus U$이면 $s - t$ 절단면이라고 한다.

위와 같이 절단면으로 분리하고 싶은 마디들을 *터미널*이라고 부르자.

정의 4.2

k-터미널절단(k-terminal cut 또는 multyway cut)문제
입력: 무향 그래프 $G = (V, E)$, 비음 호 비용 c, 터미널 집합 $\{s_1, s_2, \dots, s_k\} \subseteq V$.
출력: 제거하면 모든 터미널들이 서로 분리되는 호 집합, 즉, 터미널절단면(terminal cut) 중에서 최소비용을 갖는 것.

- $k \geq 3$이면 NP-hard.

정의 4.3

k-절단(k-cut)문제
입력: 무향 그래프 $G = (V, E)$, 비음 호 비용 c.
출력: 제거하면 모든 G가 k 개의 요소로 분리되는 호 집합, 즉, k-절단면(terminal cut) 중에서 최소비용을 갖는 것.

- k가 고정된 상수이면 다항시간에 풀 수 있음. 그렇지 않으면 NP-hard.

k-터미널절단면 알고리듬

1. 각 $i = 1, 2, , \ldots, k$에 대해 터미널 s_i를 다른 모든 터미널로부터 고립시키는 최소비용 절단면 C_i를 구한다.
2. C_i들 중에서 가장 비용이 큰 것, say, C_k를 제외시키고 나머지 절단면의 합집합 $C \equiv \bigcup_{i=1}^{k-1} C_i$를 해로 한다.

- s_i를 고립시키는 최소절단면은, $S \setminus s_i$를 하나의 마디로 축약한 그래프에서, 축약 마디와 s_i를 분리하는 최소비용 절단면. 따라서 최대흐름 알고리듬을 사용하면 다항시간에 구할 수 있다.

정리 4.4

k-터미널절단면 알고리듬은 $(2 - \frac{2}{k})$-근사해를 보장한다.

증명: 최적해를 A라고 하자. 모든 호비용이 비음이므로, A는 G를, s_i를 한 개씩 포함하는 정확히 k개의 요소로 분리한다고 가정할 수 있다. 이 때, 터미널 s_i를 포함하는 요소와 다른 요소들 사이에 걸쳐있는 A의 호들을 A_i라고 쓰자. A의 각 호는 두 개의 요소에 걸쳐있으므로

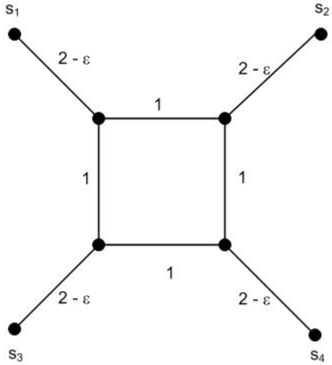
$$A = \bigcup_{i=1}^{k} A_i, \; \sum_{i=1}^{k} c(A_i) = 2c(A).$$

$c(C_i) \leq c(A_i)$이므로 C가 2-근사라는 것을 쉽게 알 수 있지만, 조금 더 나은 분석이 가능하다:

$$\begin{aligned} c(C) \leq \textstyle\sum_{i=1}^{k} c(C_i) - c(C_k) \leq \textstyle\sum_{i=1}^{k} c(A_i) - c(C_k) \\ \leq 2c(A) - \tfrac{2}{k}c(A). \end{aligned}$$

마지막 부등호는 $c(C_k)$가 어떤 $c(A_i)$보다도 크기 때문이다. □

tight example



알고리듬으로 얻은 해의 비용은

$$(2-\epsilon)(k-1) = (2-\epsilon)\left(1-\frac{1}{k}\right)k = (2-\epsilon)\left(1-\frac{1}{k}\right)\mathrm{OPT}\ \forall \epsilon > 0.$$

정의 4.5

고모리-후 나무 호비용 c를 가진 **무향**그래프 $G = (V, E)$가 있을 때, 마디집합 V와 호가중치 w를 가지는 나무 T가 다음의 성질을 만족할 때 *고모리-후 나무*라고 한다.
i) 모든 $e \in E$에 대해 T에서 호 e를 제거하면 양분되는 마디집합 U와 $V \setminus U$로 결정되는 G의 절단면의 비용이 w_e가 된다: $c(U; V \setminus U) = w_e$.
ii) 모든 마디 쌍 $u, v \in V$에 대해, 최소비용 $u - v$ 절단면의 값이 G와 T에서 같다.

- 최대흐름 알고리듬을 $n - 1$번 적용하면 고모리-후 나무는 얻을 수 있다(Exer. 4.3-4.6)!
- 이는 $\binom{n}{2}$개 마디 쌍을 분리하는 최소절단면들을 T의 $n - 1$개의 절단면으로 모두 나타낼 수 있다는 의미이다.

k-절단면 알고리듬

1. G의 G-H 나무 T를 구한다.
2. T의 호 중에서 가중치가 가장 작은 $k-1$개의 호를 골라, G에서 이에 대응하는 절단면의 합집합 C를 구한다.,

정리 4.6

위의 알고리듬은 $(2-\frac{2}{k})$-근사해를 보장한다.

증명: 우선 C가 실제로 k개 이상의 요소로 G를 분리한다는 것을 보자. (k보다 크면 k개가 되도록 C에서 호를 빼면 된다.) T에서 $k-1$의 호를 제거하면, 정확히 k개의 요소와 이 요소들에 대응하는 k개의 마디집합이 생기는데, 이 마디 집합들은 모두 G에서도 서로 분리된다는 것을 알 수 있다. 따라서 k개 이상의 요소로 G가 분리된다.

k-절단면 문제의 최적해를 A라고 하자. A를 제거할 때 생기는 k개의 요소의 마디집합을 V_1, $V_2, \ldots, V_k$, 그리고 V_i를 다른 마디들과 분리하는 절단면을 A_i라고 하자. 그러면

$$\sum_{i=1}^{k} c(A_i) = 2c(A).$$

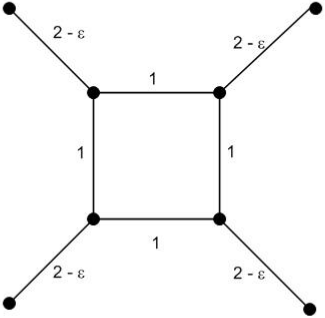
이 중에서 호비용이 가장 큰 것을 A_k라고 하자.

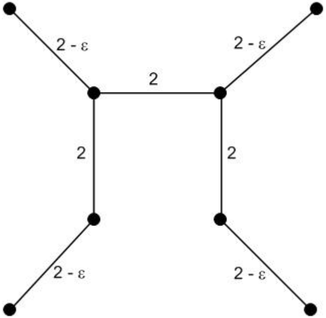
만약 T에서 가중치가 각각 $c(A_i)$ $i = 1, 2, \ldots, k-1$보다 크지 않은 $k-1$개의 호를 찾을 수 있다면 증명은 끝나게 된다. 이를 위해 T에서 V_1, $V_2, \ldots, V_k$의 각 마디집합을 하나의 마디로 축약한다. 이 때 마디집합 사이에서 남아있는 호의 집합을 B라고 하자. 만약 이렇게 얻은 그래프가 나무가 아니면 나무가 될 때까지 호를 제거한다. 남은 $k-1$개의 호집합을 B'이라고 하자. 만약 이 나무에서 $(u, v) \in B'$이고 (u, v)가 V_i에 대응하는 마디에 달려있다고 하자. 그러면 w_{uv}는 최소 $u-v$ 절단면의 비용이고 A_i는 $u-v$ 절단면 중 하나이기 때문에 $w_{uv} \le c(A_i)$가 성립한다.

C는 T의 호중에서 비용이 가장 작은 $k-1$개의 호의 비용이기 때문에,

$$c(C) \leq \sum_{e \in B'} w_e \leq \sum_{i=1}^{k-1} c(A_i)$$
$$\leq \left(1 - \frac{1}{k}\right) \sum_{i=1}^{k} c(A_i) = 2\left(1 - \frac{1}{k}\right) c(A).\square$$

tight example





Part IV

Chapter 5, 6

k-센터문제

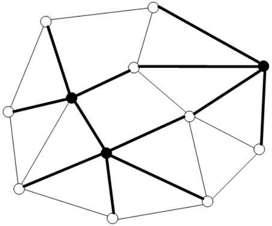
정의 5.1

메트릭 k-센터문제
입력: 삼각부등식을 만족하는 호비용 c_{uv}, $uv \in E$를 가진 무향 완전호그래프 $G = (V, E)$, 자연수 $k(< |V|)$.
최적해: 크기가 k인 V의 부분집합 S 중에서, V의 각 마디에서 S의 최소 비용 이웃 마디까지의 비용의 최대값이 최소가 되는 것:
$\min_{S \subseteq V : |S| = k} \max_{v \in V} \min_{u \in S} c_{vu}$.

정의 5.2

Dominating Set 무향그래프 $G = (V, E)$의 모든 마디의 이웃을 포함하고 있는 집합 $D \subseteq V$를 G의 dominating set이라고 한다.

이것은 G의 마디집합이 $|D|$개의 별(K_{1p}, $p \geq 1$)로 커버된다는 것과 동치이다.
(별이 모두 마디 교집합이 없다고 가정할 수 있다.)



- k-센터문제는 메트릭이 아닌 경우, 어떤 근사계수 $\alpha(n)$도 불가능.
- 메트릭 k-센터문제는 어떤 $\epsilon > 0$에 대해서도 $(2 - \epsilon)$-근사가 불가능하다.
- Hochbaum과 Shmoys는 2-근사해법 개발.

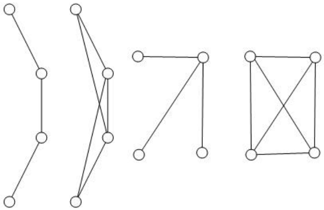
메트릭 k-센터문제의 다른 관점

호를 비용 증가 순서로 정열한다: $c_{e_1} \leq c_{e_2} \leq \cdots \leq c_{e_m}$. $E_i \equiv \{e_1, e_2, \ldots, e_i\}$. $G_i \equiv (V, E_i)$로 정의한다. k-센터문제는 크기가 k를 넘지 않은 dominating set을 가진 G_i중에서 최소 값 i^*를 선택하는 문제와 동치이다. 그러면 $c_{e_{i^*}}$가 최적목적함수 값이 된다.

- 최소 크기 dominating set을 구하는 문제는 NP-hard이다.

정의 5.3

제곱그래프(square graph) 무향그래프 $G = (V, E)$의 제곱그래프 G^2는 같은 마디집합 V와, G에서 길이 2이하의 경로로 연결되는 마디 간의 호 집합으로 주어지는 그래프로 정의한다.



기본정리 5.4

G^2의 안정집합(stable set, independent set), I의 크기는 G의 dominating set D의 크기 보다 같거나 작다.

증명: G의 마디집합은 $|D|$개의 별로 커버된다. G의 별은 G^2의 클릭이다. 한개의 클릭에서 최대 한개의 안정집합의 마디가 선택될 수 있다. 따라서, $|I| \leq |D|$. $\square$

메트릭 k-센터 알고리듬

1. $G_1^2, G_2^2, \ldots, G_m^2$를 구성한다.
2. G_i^2의 maximal 안정집합 M_i를 구한다.
3. $|M_i| \leq k$인 최소 i를 j라고 하자.
4. M_j를 해로 출력.

기본정리 5.5

$c_{e_j} \leq \mathrm{OPT}.$

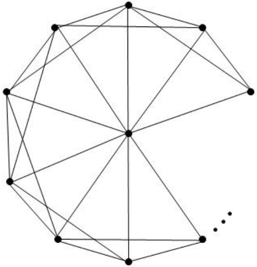
증명: $i < j$이면 $k < |M_i| \leq$ 모든 dominating set의 크기. 따라서 $i < i^*$. 즉, $c_{e_j} \leq \mathrm{OPT}$.

정리 5.6

위의 알고리듬은 2-근사해를 보장한다.

증명: 우선 M_j는 maximal 안정집합으로써, G_j^2 따라서 G의 dominating set이 된다. 이 때, 삼각부등식에 의해 G에서 센터 M_j의 마디들이 사용하는 호는 G_j^2 호 비용 2배를 넘지 않는다. 따라서, 기본정리 5.5 에 의해 정리가 성립. □

Tight example



정리 5.7

P $\neq$ NP라면, 어떠한 $\epsilon > 0$에 대해서도 메트릭 k-센터의 $(2-\epsilon)$-근사해법은 불가능하다.

증명: $G = (V, E)$에 $|D| \leq k$인 dominating set D가 존재하는가라는 결정문제를 보자. G에 호를 첨가하여 완전호 그래프로 만들자. 원래 호에는 1, 새로 첨가된 호에는 2의 비용을 준다. 그리고 이 그래프에서 정의된 메트릭 k-센터 문제에 $(2-\epsilon)$-근사해법을 적용한다.

답이 "예" 인 문제는 목적함수가 최대 $(2-\epsilon)$ 인 해를 근사해법이 구할 것이고, "아니오" 인 문제는 목적함수가 최소 2 인 해를 구할 것이다. □

정의 5.8

마디 가중치 메트릭 k-센터문제
입력: 삼각부등식을 만족하는 호비용 c_{uv}, $uv \in E$과 마디 가중치 $w : V \to \mathbb{Q}_+$를 가진 무향 완전호그래프 $G = (V, E)$, 상수 $k \in \mathbb{Q}_+$.
최적해: 가중치의 합이 k를 넘지 않는 V의 부분집합 S 중에서, V의 각 마디에서 S의 최소 비용 이웃 마디까지의 비용의 최대값이 최소가 되는 것:
$\min_{S \subseteq V : w(S) \leq k} \max_{v \in V} \min_{u \in S} c_{vu}$.

$\mathrm{wdom}(H) \equiv$ 마디 가중치를 가진 그래프 H의 dominating set의 가중치 중에서 최소값.

기본정리 5.9

그래프 H에서 마디 u의 최소 가중치 이웃 마디를 $s(u)$, I를 H^2의 안정집합이라고 할 때,

$$w(\{s(u) : u \in I\}) \leq \mathrm{wdom}(H).$$

증명: D를 H의 최소가중치 dominating set이라고 하자. 그러면 H의 모든 마디를 커버하는 D의 각 마디를 중심으로 하는 서로 마디 교집합이 없는 별이 $|D|$ 개 존재하며 이들은 H^2에서는 모두 클릭이 된다. 따라서 I는 각 별에서 한개 이상의 마디를 가질 수가 없다. 그리고 I의 마디들은 어떤 별에 속하고 따라서 그 별의 중심 마디를 이웃마디로 갖는다. 따라서, $w(\{s(u) : u \in I\}) \leq \mathrm{wdom}(H)$. □

$s_i(u) \equiv G_i$에서 마디 u의 최소 가중치 이웃마디

마디 가중치 메트릭 k-센터 알고리듬

1. $G_1^2, G_2^2, \ldots, G_m^2$를 구성한다.
2. G_i^2의 maximal 안정집합 M_i를 구한다.
3. $S_i = \{s_i(u) : u \in M_i\}$를 구한다.
4. $w(S_i) \leq k$인 최소 i를 j라고 하면, S_j를 해로 한다.

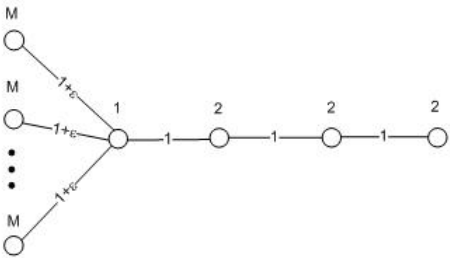
정리 5.10

3-근사

증명: 마디 가중치가 없는 경우와 같이, 기본정리 5.9에 의하여, $c_{e_j} \leq \mathrm{OPT}$임을 알 수 있다.

모든 마디 v는 M_j의 어떤 마디 u와 G_j에서 두 개 이하의 호를 가진 경로로 연결된다. 또한 $u \in M_j$와 $s_j(u)$는 정의에 따라, G_j의 한개의 호로 연결된다. 모든 G의 마디와 S_j의 어떤 마디는, 따라서, G_j의 호를 세개 이하 가진 경로로 연결된다. 삼각부등식에 의해 그 경로의 길이는 $\leq 3c_{e_j} \leq 3\mathrm{OPT}$가 된다. □

Tight example



피드백 마디집합

정의 6.1

피드백 마디집합
입력: 마디 가중치 $w : V \to \mathbb{Q}_+$를 가진 무향 그래프 $G = (V, E)$.
최적해: 제거하면 G를 무회로(acyclic) 그래프로 만드는 마디 집합 중에서 가중치의 합이 최소가 되는 것.

- 제거하면 G를 무회로(acyclic) 그래프로 만든다는 것은 G의 모든 회로와 교차한다는 것과 같다.

$\mathrm{GF}[2] \equiv$ 체(field)로서의 $\{0, 1\}$,
참고: 체(field) = 가환나누기환(commutative division ring)

$$0 + 0 = 0,\ 0 + 1 = 1 + 0 = 1,\ 1 + 1 = 0,$$

$$0 \cdot 0 = 0 \cdot 1 = 1 \cdot 0 = 0,\ 1 \cdot 1 = 1$$

G의 호의 개수 만큼의 차원을 가진 벡터 공간 $\{0,1\}^{|E|}$에서, G의 단순회로 C의 특성벡터 χ^C들로 생성된(span) 부분공간, 즉, *회로공간*의 차원을 *cyclomatic number*, $\mathrm{cyc}(G)$라고 한다. 회로공간의 원소들은 모든 마디 차수가 2인 부분그래프가 되는 것을 알 수 있다.

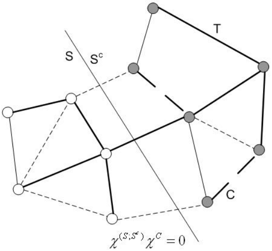
정리 6.2

$$\mathrm{cyc}(G) = |E| - |V| + \kappa(G).$$

우선 G의 회로공간은 각 연결요소의 회로공간의 합(direct sum)이 된다. 따라서, G가 연결된 그래프라고 가정하고 정리를 증명하면 된다: $\mathrm{cyc}(G) = |E| - |V| + 1$.

T를 G의 걸침나무라고 하자. T밖의 각 호 e가 T의 호와 만드는 회로들은 모두 선형독립인 특성벡터를 만든다. 따라서, $\mathrm{cyc}(G) \geq |E| - (|V| - 1)$.

한편 T의 각 호에 의해 정의되는 절단면 $(S; S^c)$의 특성벡터는, 모든 회로의 특성벡터와 서로 수직이다. 즉 내적이 0이 된다. 왜냐하면, 절단면과 회로는 항상 짝수 개의 호들을 공유하기 때문이다.



이러한 절단면은 $|V| - 1$개 존재하는데 모두 선형독립이다. 즉, 회로 공간의 여공간의 차원은 최소 $|V| - 1$이 되며, 따라서 회로 공간의 차원은 $|E| - (|V| - 1)$을 넘지 않는다. □

연결된 그래프 G에서 마디 v를 제거할 때, cyclomatic number의 감소를 $\delta_G(v)$라고 하자. 정리 6.2를 G와 $G \setminus v$에 적용하면,
$\mathrm{cyc}(G) = |E| - |V| + 1$, 그리고
$\mathrm{cyc}(G \setminus v) = |E| - \deg_G(v) - |V| + 1 + \kappa(G \setminus v)$를 얻는다. 이로부터, 연결된 그래프 G에서는

$$\delta_G(v) = \deg_G(v) - \kappa(G \setminus v). \tag{6.1}$$

기본정리 6.3

H가 G의 부분 그래프이면 $\delta_H(v) \leq \delta_G(v)$.

증명: v를 제거할 때, 영향을 받는 부분은 이를 포함하는 요소만이다. 따라서, G와 H가 연결되었다고 가정하자. (6.1)에 의하여 다음을 증명하는 것과 같다.

$$\deg_H(v) - \kappa(H \setminus v) \leq \deg_G(v) - \kappa(G \setminus v).$$

양쪽의 항들을 비교하자. 우선 $\deg_H(v) \leq \deg_G(v)$이다. H에서 v를 제거하여 생기는 요소들을 $c_1, c_2, \ldots, c_k$라고 하자. G에만 있는 호 중에서 v에 달려 있지 않은 호는 이 요소들을 연결시켜 개수를 감소 시키는데 만 도움이 된다. 따라서, 만약 $G \setminus v$가 요소를 더 가지고 있다면, 이는 제거되는 v에 달려 있으며 G에만 있는 호 때문에 추가 되는 요소들이다. 그러나, 이러한 요소 한 개 당 $\deg_G(v)$가 $\deg_H(v)$보다 최소 1 크다. 따라서 부등호가 성립.$\square$

만일 $F = \{v_1, v_2, \ldots, v_f\}$가 피드백마디집합이면 F를 제거하면 cyclomatic number는 0이 되므로, $G_0 \equiv G$, $G_i \equiv G \setminus \{v_1, \ldots, v_i\}(i = 1, 2, \cdots)$라고 하면,

$$\mathrm{cyc}(G) = \sum_{i=1}^{f} \delta_{G_{i-1}}(v_i).$$

기본정리 6.3에 의하여,

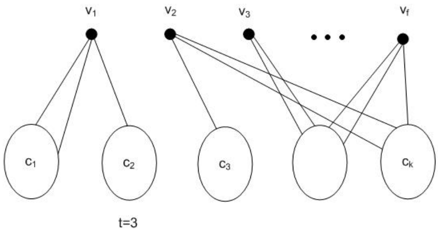
$$\mathrm{cyc}(G) \leq \sum_{e \in F} \delta_G(v). \tag{6.2}$$

기본정리 6.4

F가 minimal 피드백 마디집합이면 $\sum_{v \in F} \delta_G(v) \leq 2\mathrm{cyc}(G)$.

증명: 역시 G가 연결되었다고 가정할 수 있다. $F = \{v_1, v_2, \ldots, v_f\}$ 그리고 F를 제거하면 k개의 요소가 생긴다고 하자. 이중 G에서 F의 오직 한개의 마디에 호로 연결되는 요소들이 t개라고 하면,

$$\sum_{i=1}^{f} \kappa(G - v_i) = f + t. \tag{6.3}$$



정리 6.2에 의하여 $\sum_{i=1}^{f} \delta_G(v_i) = \sum_{i=1}^{f}(\deg_G(v_i) - \kappa(G - v_i))$, $\mathrm{cyc}(G) = |E| - |V| + 1$이므로, 만약 다음을 보이면 증명은 끝난다:

$$\sum_{i=1}^{f}(\deg_G(v_i) - \kappa(G - v_i)) \leq 2(|E| - |V|).$$

이것은 (6.3)에 의하여 다음과 동치이다:

$$\sum_{i=1}^{f} \deg_G(v_i) \leq 2(|E| - |V|) + f + t.$$

f개의 마디를 제거하여 얻은 k개의 각 요소는 나무가 된다, 따라서 요소 내부의 호의 수를 합하면

$$|V| - f - k. \tag{6.4}$$

절단면 $(F; V \setminus F)$의 호의 개수의 하한을 구해보자. F가 minimal이므로 각 v_i는 F의 다른 마디를 포함하지 않는 회로에 포함되어야 한다. 따라서 (위의 그림과 같이) 각 v_i는 두 개 이상의 호로 연결되는 요소를 최소 하나 가지고 있어야 한다. 각 v_i에 대하여 이 두 개의 호 중 하나를 제거하면 모두 f개의 호가 제거된다. 마디와 요소 한 쌍을 중복 연결하는 호를 한 개씩 제거하였으므로, 이렇게 f개를 제거한 후에도, t개의 요소들은 여전히 F의 마디 중에 최소 한개와 호로 연결되어야 하며, $k-t$개의 요소들은 최소 두 개와 호로 연결되어야 한다. 따라서 $(F; V \setminus F)$의 호의 개수는 최소

$$f + t + 2(k - t) = f + 2k - t. \tag{6.5}$$

따라서, (6.4)과 (6.5)에 의하여

$$\begin{aligned} \textstyle\sum_{i=1}^{f} \deg_G(v_i) &\le 2|E| - 2(|V| - f - k) - (f + 2k - t) \\ &= 2(|E| - |V|) + f + t. \square \end{aligned}$$

정의 6.5

어떤 상수 $c > 0$에 대해 각 마디 v의 가중치가 $w_c = c\delta_G(v)$를 만족하면 w를 cyclomatic이라고 부른다.

(6.2)에 의하여 w가 cyclomatic이면 $c\mathrm{cyc}(G) \leq \mathrm{OPT}$ 이다. 또한 기본정리 6.4에 의하여 w가 cyclomatic이고 F가 minimal 피드백마디집합이면 $w(F) \leq 2c\mathrm{cyc}(G)$이다:

$$c\mathrm{cyc}(G) \leq \mathrm{OPT} \leq w(F) \leq 2c\mathrm{cyc}(G).$$

따라서, 가중치 w가 cyclomatic이면 임의의 minimal 피드백집합 F에 대해

따름정리 6.6

$$w(F) \leq 2\mathrm{OPT}.$$

이것을 일반적인 가중치에 적용하기 위하여 다음과 같은 'layering'을 생각하자.
$c \leftarrow \min_{v \in V} \left\{ \frac{w_v}{\delta_G(v)} \right\}$; $t_v \leftarrow c\delta_G(v)$; $w'_v \leftarrow w_v - t_v$.

또한 *잔여가중치* w'_v가 양수인 마디집합을 $V'(\subsetneq V)$, G'을 V'으로 유도된 부분그래프라고 하자.

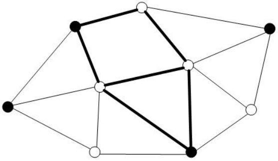
이러한 과정을 G 대신 G'에, 무회로 그래프가 될 때까지, 반복한다:
$G = G_0 \supsetneq G_1 \supsetneq \cdots \supsetneq G_k$
$V = V_0 \supsetneq V_1 \supsetneq \cdots \supsetneq V_k$.

또한, $t^i(1 = 0, 1, \ldots)$는 G_i에서 정의된 cyclomatic 가중치, $w^i(1 = 0, 1, \ldots)$는 G_i의 잔여가중치라고 하자: $w^0 \equiv w$, $w^1 \equiv w - t^0$, ... 그리고 편의상 $t^k = w^k$라고 놓자. 그러면,

$$\sum_{i: v \in V_i} t^i_v = w_v. \tag{6.6}$$

기본정리 6.7

H를 V'으로 유도된 $G = (V, E)$의 부분그래프라고 하자. F를 H의 minimal 피드백 마디집합, $F' \subseteq V \setminus V'$을 $F \cup F'$이 G의 피드백 마디집합이 되는, minimal 집합이라고 하자. 그러면 $F \cup F'$이 G의 minimal 피드백 마디집합이 된다.



이런 G의 분해과정에서 무회로인 G_k의 minimal 피드백마디집합은 $F_k = \emptyset$이 된다. 이를 초기해로 하여 $i = k, k-1, \ldots, 1$에 대하여, G_i의 minimal 피드백 집합을 $V_{i-1} - V_i$의 minimal 집합을 사용하여 G_{i-1}의 minimal 피드백 집합 F_{i-1}로 확장한다. 이를 반복하여 F_0를 해로 한다.

정리 6.8

2-근사.

증명: F^*를 최적해라고 하자. 그러면 $F^* \cap V_i$는 G_i의 피드백집합이된다. 그리고 t^i**에 대한** G_i의 피드백 집합의 최소가중치를 OPT_i라고 하면,

$$\mathrm{OPT} = w(F^*) = \sum_{i=0}^{k} t^i(F^* \cap V_i) \geq \sum_{i=0}^{k} \mathrm{OPT}_i.$$

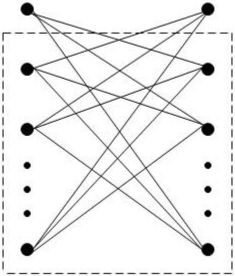
또한 F_0 역시 다음과 같이 분해할 수 있다:

$$w(F_0) = \sum_{i=0}^{k} t^i(F_0 \cap V_i) = \sum_{i=0}^{k} t^i(F_i).$$

기본정리 6.7에 의하여, F_i는 G_i의 minimal 피드백집합이다. 그리고 모든 $0 \leq i \leq k-1$에 대해 t_i가 cyclomatic 가중치이므로 $t_i(F_i) \leq 2\mathrm{OPT}_i$. 따라서,

$$w(F_0) \leq 2\sum_{i=0}^{k} \mathrm{OPT}_i \leq 2\mathrm{OPT}. \square$$

Tight example



minimal 피드백 집합을 구하는 방법

V의 부분집합 U로 유도된 그래프 $G[U]$를 생각하자. $U \leftarrow \emptyset$으로 시작하여, $G[U]$에 회로가 발생할 때 까지 V의 마디를 첨가한다. 그러면 $F := V \setminus U$는 minimal 피드백 집합이 된다.

이 방법은 기본정리 6.7에서 F와 F'을 구하는데 자연스럽게 적용할 수 있다: 위의 방법으로 H에서 F를 구한 후에 이를 G에 확장시킬 때는 $V \setminus V'$의 마디 중에서 회로가 발생되지 않을 때까지 첨가한다. 이 중 첨가되지 않은 마디들의 집합이 F'이 된다.

참고 6.9

유향그래프에서는 $O(\log|V| \log\log|V|)$-근사 가능 (Seymour[95], Even et al[95]).

Part V

Chapter 8-10

완전다항근사해법군(FPTAS)

정의 7.1

근사해법군 최적화문제 Π의 가능해 x의 목적함수를 $f_\Pi(x)$라고 하자. 다음과 같은 알고리듬 $\mathcal{A}$를 *근사해법군* 이라고 부른다: 임의의 문제 예 I와 $\epsilon > 0$을 입력하면 다음을 만족하는 가능해 x를 출력한다.

- Π가 최대화문제이면 $f_\Pi(x) \geq (1-\epsilon)\mathrm{OPT}$,
- 최소화문제이면 $f_\Pi(x) \leq (1+\epsilon)\mathrm{OPT}$.

$\mathcal{A}$의 수행시간이 고정된 $\epsilon > 0$에 대하여 다항식이면 다항시간근사해법군(PTAS, polynomial time approximation scheme)이라고 한다. 더나아가, $\frac{1}{\epsilon}$과 Π 입력크기의 다항식 이면 완전다항시간근사해법군 (FPTAS, fully polynomial time approximation scheme)이라고 한다.

문제 7.2

배낭문제 부피 $w_j \in \mathbb{Z}_+$, 효용 $p_j \in \mathbb{Z}_+$를 가진 n개의 품목, $j \in N = \{1, 2, \ldots, n\}$을 부피 $W \in \mathbb{Z}_+$를 가진 배낭에 효용의 합이 최대가 되도록 선택하여 넣는다.

<u>**배낭문제를 위한 유사다항 알고리듬**</u>

$\langle I \rangle \equiv$ 문제 예 I의 이진 입력크기

$|I| \equiv$ 문제 예 I의 unary 입력크기

정의 7.3

문제 Π의 모든 문제 예 I를, 어떤 다항함수 p에 대하여 $p(|I|)$-시간에 푸는 해법을 유사다항시간 해법이라고 부른다.

알고리듬 7.4

유사다항알고리듬 I: $z(W', l)$ =품목 $\{1, 2, \ldots, l\}(l \leq n)$로 부피 $W'(\leq W)$을 가진 배낭에서 얻을 수 있는 최대 효용. ($W' \geq w_1$이면 $z(W', 1) = p_1$, $0 \leq W' < w_1$이면 $z(W', 1) = 0$으로 초기화. $W' < 0$이면 $z(W', l) = -\infty$로 정의.

$$z(W', l) = \max\{z(W' - w_l, l-1) + p_l, z(W', l-1)\}. \tag{7.7}$$

알고리듬 7.5

유사다항알고리듬 II: $w(Z, l)$ =품목 $\{1, 2, \ldots, l\}(l \leq n)$로 효용 Z를 얻기 위해 필요한 배낭의 최소 부피 ($\Rightarrow w(0, l) = 0$, $l = 1, 2, \ldots, n$.) $Z < 0$이면 $w(Z, l) = \infty$로 정의.

$$w(Z, l) = \min\{w(Z - p_l, l-1) + w_l, w(Z, l-1)\}. \tag{7.8}$$

$\mathrm{OPT} = w(Z, n) \leq B$인 최대 Z.
$O(n^2 \max p_j)$ 시간 알고리듬.

배낭문제를 위한 FPTAS

아이디어: 데이터의 정확도를 낮추어 (낮은 자리 수자를 일정부분 무시) 입력크기를 n과 $\frac{1}{\epsilon}$(ϵ 오차)의 다항식으로 한다.

$P \equiv \max\{p_j\}$, $\hat{p}_j = \lfloor \frac{p_j}{\epsilon P/n} \rfloor$로 문제를 근사하여 유사다항알고리듬 II를 적용하여 구한 최적해(품목집합) S를 근사해로 사용.

성질 7.6

$$p(S) \geq (1-\epsilon)\mathrm{OPT}.$$

증명: 우선 모든 집합 $S \subseteq N$에 대해 $\hat{p}(S) \geq \sum_{i \in S}(\frac{p_j}{\epsilon P/n} - 1) \equiv \frac{1}{\epsilon P/n} p(S) - |S|$.
따라서, $(\epsilon P/n)\hat{p}(S) \geq p(S) - (\epsilon P/n)|S| \geq p(S) - \epsilon P \geq p(S) - \epsilon \mathrm{OPT}$.
따라서, 원래 문제의 최적해를 S^*라고 하면 $(\epsilon P/n)\hat{p}(S^*) \geq (1-\epsilon)\mathrm{OPT}$.

그런데, $p(S) \geq (\epsilon P/n)\hat{p}(S) \geq (\epsilon P/n)\hat{p}(S^*)$.
따라서, $p(S) \geq (1-\epsilon)\mathrm{OPT}$. □

위의 알고리듬의 수행시간은 $O(n^2 \lfloor \frac{P}{\epsilon P/n} \rfloor) = O(n^2 \lfloor \frac{n}{\epsilon} \rfloor) = O(\frac{n^3}{\epsilon})$. 따라서 배낭문제를 위한 FPTAS가 된다.

Strong NP-hardness & FPTAS

정의 8.1

NP의 모든 문제가, 입력데이터가 unary로 표현된 Π로 다항변환이 가능하면, Π를 *strongly* NP-*hard*라고 한다.

정의 8.2

문제 Π의 모든 문제 예 I가, 어떤 다항함수 p가 존재하여, $|I| \le p(\langle I \rangle)$를 만족하면 Π를 *비수치문제* (no number problem)이라고 한다. 그렇지 않으면 *수치문제* (number problem)라고 한다.

$$\Pi_p = \{I \in \Pi : |I| \le p(\langle I \rangle)\}.$$

성질 8.3

만약 Π가 NP-*complete*이고 비수치문제라면 유사다항시간 알고리듬은 $\mathrm{P} \neq \mathrm{NP}$인 한 불가능하다.

정의 8.4

만약 어떤 다항함수 p에 대해 Π_p가 NP-complete(NP-hard)이면 Π를 *strongly* NP-*complete* (*strongly* NP-*hard*)라고 한다.

정리 8.5

p를 다항함수 Π를 NP-hard 최소화 최적화문제라고 하자. 목적함수 f_Π는 정수이고 모든 문제 예 I에 대해 $\mathrm{OPT}(I) < p(|I|)$가 성립한다고 하자. 만약 Π가 FPTAS를 가지면 유사다항 알고리듬을 갖는다.

증명: $\epsilon = 1/p(|I|)$. □

따름정리 8.6

Π가 정리 8.5의 조건을 만족한다고 하자. 만약 Π가 strongly NP-hard이면 FPTAS는 불가능하다.

Part VI

Chapter 11

직선거리(Euclidean) TSP

d-차원 공간의 두 점 x, y 사이에 다음과 같은 직선거리를 생각하자: $\sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$.

정의 9.1

직선거리(Euclidean) TSP: d-차원 직선거리 공간(Euclidean space)에 n개의 점이 있을 때, 이들을 포함하는 직선거리의 합이 가장 작은 순회를 구한다.

우리는 $d = 2$인 경우, 즉 평면 위의 문제를 가정한다. (일반적인 경우로 쉽게 확장할 수 있다.)

우선, 모든 점들을 포함하는 가장 작은 정사각형의 길이를 L이라고 하면, L은 $4n^2$, 그리고 정사각형의 간격이 1인 격자 위에 모든 n개의 점들이 위치하고 있다고 가정할 수 있다 (n은 2의 거듭 제곱이라고 가정.):
$L = 4n^2$, $L = 2^k$, $k = 2 + 2\log_2 n$.

또한 m은 구간 $[\frac{k}{\epsilon}, \frac{2k}{\epsilon}]$에 속하는 2의 거듭 제곱수라고 하자.
$\Rightarrow \; m = O(\frac{\log n}{\epsilon})$.

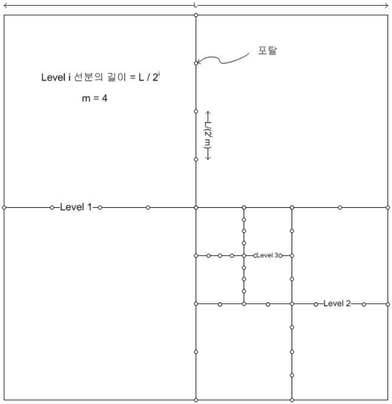
기본분할

$$L \times L \rightarrow 4\,\frac{L}{2} \times \frac{L}{2}\ \text{(level 1)} \ \rightarrow \cdots \rightarrow 4^i\,\frac{L}{2^i} \times \frac{L}{2^i}\ \text{(level i)} \ \rightarrow \cdots$$

이러한 기본분할은 모든 마디가 자식을 네개씩 가지는 나무 T로 표시할 수 있다. 이때 나무의 각 마디는 물론 한 개의 정사각형을 의미한다.

포털

순회가 정사각형을 들고 날때는 항상 포털을 경유한다.



정의 9.2

τ가 n개의 점과 어떤 포털 부분집힙으로 이루어진 순회 (tour)이며, 포털을 제외한 나머지 부분에서는 호의 교차가 발생하지 않을 때, *기본분할에 맞는 순회*라고 부르자.
기본분할에 맞는 순회 τ가 어떤 포털도 두번을 초과하여 경유하지 않을 때, 기본분할에 맞는 *교차가 제한된 순회*라고 부르자.

기본정리 9.3

τ를 기본분할에 맞는 순회라고 하면, 길이가 τ를 넘지 않는 교차가 제한된 순회를 구할 수 있다.

증명: 양편에서 Short-Cutting. □

기본정리 9.4

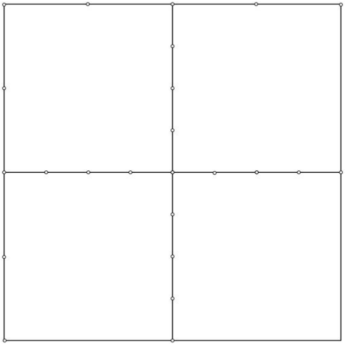
기본 분할에 맞는 제한된 교차 순회 중에 최적 순회를 $2^{O(m)} = n^{O(\frac{1}{\epsilon})}$ *시간 안에 구할 수 있다.*

증명 스케치: 동적계획법에 의하여 나무 T의 각 정사각형의 모든 "valid visit"의 비용의 표를 유지해 간다. T의 깊이가 $k = O(\log n)$이므로 정사각형의 개수는 n의 다항함수 (계산해 볼것).

τ를 기본 분할에 맞는 제한된 교차 순회 중에 최적 순회라고 하자. τ가 나무 T의 어떤 정사각형을 들고 나는 총횟수는 $8m$을 넘지 않는다. τ 중에서 S에 속하는 부분은, S의 네개의 변의 포털들에서 시작하고 끝나는 최대 $4m$개의 경로로 이루어 진다. 이렇게 네변의 포털들은 같은 경로의 시작과 끝이 됨에 따라 짝을 지울 수 있는데, 사용된 포털의 집합 그 짝짓기를 *valid visits*이라고 부른다.

각 포털은 0, 1, 그리고 2 번까지 사용될 수 있으므로 전체 $3^{4m} = n^{O(\frac{1}{\epsilon})}$의 가능성이 있지만 이중에서 짝수, $2r$개의 포털이 들고 날때 사용되어 짝지워지는 경우만 생각하면 된다. 이는 2^{2r}을 넘지 않으며 그 수는 역시 $n^{O(\frac{1}{\epsilon})}$를 넘지 않는다. 따라서 S의 valid visit의 수는 $n^{O(\frac{1}{\epsilon})}$.

한 개의 valid visit의 최적 비용은 어떻게 구할까? S가 i번째 레벨의 마디라고 하자. 그리고 임의의 valid visit V를 생각하자. S는 레벨 $i+1$에 네개의 정사각형으로 분할 되며, 이 때, 네 개의 내부 직선이 최대 $4m$개의 추가 포털을 갖게 된다.

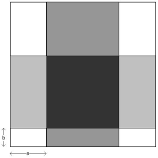


네 개의 정사각형의 모든 포털이 들고 나는 점으로 사용되는 경우의 수는 역시 $n^{O(\frac{1}{\epsilon})}$. 따라서 네 개 사각형의 모든 valid visit의 조합이 만드는 경우의 수 $n^{O(\frac{1}{\epsilon})}$이며 이들의 각 사각형에 해당하는 valid visit의 최소 비용은 이미 알고 있다.

네 개 사각형의 모든 valid visit의 조합 중에서 V와 맞는 것들 만을 고려한다. 그 중에서 비용의 합이 최소가 되는 것이 V의 최소비용이 된다. □

만약에 기본정리 9.4에서 구한, 기본 분할에 맞는 제한된 교차순회 중의 최적해의 비용이 $\leq (1+\epsilon)\mathrm{OPT}$이면 PTAS가 된다. 그러나, 그렇지 않은 예를 만들수 있다.

이러한 점을 해결하기 위해 다음과 같은 무작위로 선택된 분할을 생각하자: 기본분할을 (a, b) $(0 \leq a, b < L)$만큼 이동한 (a, b)-이동 분할을 생각하자. 즉, 기본분할에서 좌표 (x, y)를 지나는 수직선과 수평선을 각각 $(a + x) \bmod L$와 $(b + y) \bmod L$를 지나도록 이동한 분할을 생각해보자.



우리는 만약 a, b를 무작위로 선택하면 (a, b)-이동 분할에 맞는 제한된 교차 순회 중의 최적해의 비용이 $\leq (1+\epsilon)\mathrm{OPT}$가 될 확률이 $\frac{1}{2}$ 이상이 됨을 보일 것이다.

π를 최적 순회, $N(\pi)$를 π가 격자선들을 수평 또는 수직으로 교차하는 횟수라고 하자. (격자점을 교차하는 경우, 두번으로 센다.) 그러면, 다음은 쉽게 증명할 수 있다.

기본정리 9.5

$$N(\pi) \leq 2 \cdot \mathrm{OPT}.$$

정리 9.6

a, b를 $[0, L)$에서 무작위로 선택하면 π를 (a, b)-이동 분할에 맞추어도, 순회의 길이 증가의 기대 값은 $2\epsilon\mathrm{OPT}$를 넘지 않는다.

증명: π를 (a, b)-이동 분할에 맞추기 위해서는 어떤 선 l을 지날 때, 포털을 지나지 않으면 해당 선분을 두개로 나누어 가장 가까운 포털을 지나도록 해야 한다. 이때, 이로인한 순회 비용 증가는 선 l의 포털 간격을 넘지 않는다. l이 i-번째 레벨의 선이 될 확률은 $\frac{2^i}{L}$이며, 이 때 포털 간격은 $\frac{L}{2^i m}$이 된다. 따라서, 순회 비용 증가의 기대 값은

$$\sum_i \frac{L}{2^i m}\frac{2^i}{L} = \frac{k}{m}.$$

그런데, $\frac{k}{\epsilon} \le m \le \frac{2k}{\epsilon}$이므로 이 값은 ϵ을 넘지 않는다. 따라서 기본정리 9.5를 사용하면 정리가 증명된다. $\square$.

기본정리 9.7

마아코프 부등식(Markov Inequality): X가 비음 확률변수이면

$$\Pr(X \geq kE[X]) \leq \frac{1}{k}.$$

증명: $E[X] = \int_0^\infty xf(x)dx \geq c\Pr[X \geq c]\ \forall\ c > 0$. $c \leftarrow kE[X]$로 대입. □

정리 9.6에서 순회증가의 기대값은 $E[X] \leq 2\epsilon\mathrm{OPT}$. 따라서, 마아코프 부등식에서 $k = 2$로 놓으면 다음과 같은 사실을 얻는다.

따름정리 9.8

무작위로 선택한 (a, b)-이동 분할에 맞는 길이가 $(1 + 4\epsilon)\mathrm{OPT}$를 넘지 않는 순회가 존재할 확률이 최소 $\frac{1}{2}$이 된다.

Part VII

Chapter 12-15

Part II. LP-기반 근사해법

Review

- The LP-duality theorem
- Min-max relations and LP-duality

Two LP-based algorithm design techniques

- Rounding
- Primal-dual schema

Also we can use LP-duality to analyze approximation algorithms, by *dual fitting*.

Dual fitting analysis

문제 11.1

Set Cover (SC)
입력: 유한집합 U, $|U| = n$. U의 부분집합의 콜렉션 $\mathcal{S} = \{S_1, \ldots, S_m\}$, $c : \mathcal{S} \longrightarrow \mathbb{Q}_+$.
최적해: 합하면 U가 되는 $\mathcal{S}$의 최소비용 부분 콜렉션.

$$\begin{aligned} \min \textstyle\sum_{S \in \mathcal{S}} c_S x_S \\ \text{sub. to } \textstyle\sum_{S : e \in S} x_S \geq 1 \ \forall e \in U \\ x_S \in \{0, 1\} \ S \in \mathcal{S}. \end{aligned}$$

선형계획 완화는 다음과 같이 쓸 수 있다:

$$
\begin{aligned}
&\text{(fractional covering)} \\
&\mathrm{OPT}_p = \min \textstyle\sum_{S \in \mathcal{S}} c_S x_S \\
&\text{sub. to } \textstyle\sum_{S: e \in S} x_S \geq 1 \ \forall e \in U \quad \longleftrightarrow \quad y_e \\
&x_S \geq 0
\end{aligned}
\tag{11.9}
$$

$$
\begin{aligned}
&\text{(fractional packing)} \\
&\max \textstyle\sum_{e \in U} y_e \\
&\text{sub. to } \textstyle\sum_{e: e \in S} y_e \leq c_S \ \forall S \in \mathcal{S} \\
&y_e \geq 0 \ \forall e \in U
\end{aligned}
\tag{11.10}
$$

Recall

현재까지 커버된 마디들을 C라고 할 때,
S의 집합 S의 *평균비용*을 다음과 같이 정의:

$$p_e = \frac{c(S)}{|S - C|}, \ \forall e \in S - C.$$

Greedy 알고리듬
1. $C \longleftarrow \emptyset$;
2. **while** $C \neq U$ **do**
현재 가장 작은 평균비용(α라고 하자)을 갖는
S의 집합, say, S를 선택;
$S - C$의 각 원소 e들의 *가격*, $p_e = \alpha$로 정의;

$C \longleftarrow C \cup S$;
3. 선택된 집합들을 출력.

여기서 p_e를 선형완화의 쌍대문제(fractional packing)의 해로 고려하면, 항상 가능해가 되지는 않는다 (HW: 13.2).

그러나, 다음과 같이 정의하면 가능해가 된다:

$$y_e := \frac{p_e}{H_n}.$$

증명: $\mathcal{S}$의 임의의 집합 S가 k개의 원소를 가진다고 하자. 이들을 Greedy 알고리듬으로 커버되는 순서로 $e_1, e_2, \ldots, e_k$라고 하자. e_i가 커버되는 반복단계를 생각해보자. 이 순간 최소한 $k-i+1$개의 커버되지 않은 원소가 존재한다. 따라서 S는 e를 최대 $c(S)/(k-i+1)$의 평균비용으로 커버할 수 있다. 알고리듬은 가장 저렴하게 e를 커버하므로 $p_e \leq c(S)/(k-i+1)$.
따라서,

$$\begin{aligned} y_{e_i} &\leq \tfrac{1}{H_n} \cdot \tfrac{c(S)}{k-i+1} \\ \Rightarrow \textstyle\sum_{i=1}^{k} y_{e_i} &\leq \tfrac{c(S)}{H_n} \cdot \left(\tfrac{1}{k} + \tfrac{1}{k-1} + \cdots + \tfrac{1}{1}\right) \\ &= \tfrac{H_k}{H_n} \cdot c(S) \leq c(S). \ \square \end{aligned}$$

따라서, y가 쌍대 가능해가 되므로,

$$\sum_{e \in U} p_e = H_n\Big(\sum_{e \in U} y_e\Big) \leq H_n \cdot \mathrm{OPT}.$$

LP-rounding for Set Cover

$f := \mathcal{S}$의 집합들에 각 원소가 출현하는 횟수 중 최대 값.

알고리듬 12.1

LP-rounding algorithm

1. 선형완화문제의 최적해를 구한다;
2. $x_S \geq \frac{1}{f}$인 S를 모두 고른다.

정리 12.2

알고리듬 12.1은 근사계수 f를 보장한다.

증명: 임의의 $e \in U$를 생각하자. e를 포함하는 집합은 많아야 f개, 따라서 최소 한개의 집합 S는 fractional cover의 해에서 $x_S \geq \frac{1}{f}$가 되야한다. 따라서, 위의 알고리듬이 선택한 집합 S의 집합 $\mathcal{C}$는 가능해가 되어야 한다.
그리고 $x_S \geq \frac{1}{f}$인 경우에만 1이 되므로, fractional solution들은 f배를 초과하여 증가하지 않는다. 따라서, 출력된 정수해의 목적함수 값은 fractional cover의 목적함수 값 OPT_p의 f배를 넘지 않는다. □

알고리듬 12.3

확률 라운딩 알고리듬

1. 선형완화문제의 최적해 x를 구한다;

2. 각 집합이 x_S의 확률로 선택되도록 무작위로 각 집합을 선택하여 $\mathcal{C}$를 구한다.
(앞면과 뒷면이 나올 확률이 각각 x_S, $1 - x_S$인 동전을 던져 앞면이 나오면 집합 S를 해에 포함시킨다.)

3. 위의 과정을 $c \log n$ 번 반복하여 그 합집합 $\mathcal{C}'$을 해로 한다.
(c의 값은 아래에 설명.)

$$\mathbf{E}[c(\mathcal{C})] = \sum_{S \in \mathcal{S}} c_S \cdot \Pr[S \text{ 가 선택됨}] = \sum_{S \in \mathcal{S}} c_S x_S = \mathrm{OPT}_p$$

U의 각 원소 a가 $\mathcal{C}$에 의해 커버될 확률은?

$\mathcal{S}$의 k개의 집합이 원소 a를 포함한다고 하자. 그 최적해의 값들을 $x_1, x_2, \ldots, x_k$이라고 하자. 그러면 $x_1 + x_2 + \cdots + x_k \geq 1$. 이 집합 중 최소한 하나가 선택될 확률은

$$1 - (1 - x_1)(1 - x_2) \cdots (1 - x_k) \geq 1 - (1 - \frac{1}{k})^k \geq 1 - \frac{1}{e}.$$

알고리듬에서 다음이 만족되도록 c를 잡는다:

$$\left(\frac{1}{e}\right)^{c \log n} \leq \frac{1}{4n}.$$

그러면,

$$\Pr[a\text{가 } \mathcal{C}'\text{으로 커버되지 않음}] \leq \left(\frac{1}{e}\right)^{c \log n} \leq \frac{1}{4n}.$$

따라서 $\mathcal{C}'$이 가능해가 아닐 확률은 $\leq n \cdot \frac{1}{4n} \leq \frac{1}{4}$.
한편, $\mathbf{E}[c(\mathcal{C}')] \leq c \log n \mathrm{OPT}_p$이므로, 마아코프 부등식을 적용할 수 있다: X가 비음 확률변수이면, 임의의 양수 t에 대해 $\Pr[X \geq t] \leq \frac{\mathbf{E}[X]}{t}$.

$$\Pr[c(\mathcal{C}') \geq 4c \log n \cdot \mathrm{OPT}_p] \leq \Pr[c(\mathcal{C}') \geq 4\mathbf{E}[c(\mathcal{C}')]] \leq \frac{1}{4}.$$

따라서 $\mathcal{C}'$이 가능해이고 그 비용이 $4c \log n \cdot \mathrm{OPT}_p$를 넘지 않을 확률은 $\frac{1}{2}$ 이상이 된다.

$\frac{1}{2}$-정수성을 사용한 마디커버 2-근사 알고리듬.

$$\begin{aligned} \min \textstyle\sum_{v \in V} c_v x_v \\ \text{sub. to } x_u + x_v \geq 1 \ \forall (u, v) \in E \\ x_v \in \{0, 1\}, \ \ v \in V. \end{aligned} \tag{12.11}$$

$$\begin{aligned} \min \textstyle\sum_{v \in V} c_v x_v \\ \text{sub. to } x_u + x_v \geq 1 \ \forall (u, v) \in E \\ x_v \geq 0, \ \ v \in V. \end{aligned} \tag{12.12}$$

기본정리 12.4

(12.12)은 $\frac{1}{2}$-정수성을 갖는다. 즉, 그 꼭지점들은 모두 그 원소의 값이 0, $\frac{1}{2}$, 또는 1이 된다.

증명: 원소의 값이 0, $\frac{1}{2}$, 또는 1이 아닌 값을 갖는 가능해 x는 다른 두 가능해의 볼록조합으로 표시되는 것을 보이자.

$$V_+ = \left\{ v : \frac{1}{2} < x_v < 1 \right\},\ V_- = \left\{ v : 0 < x_v < \frac{1}{2} \right\}.$$

$\epsilon > 0$에 대하여, 다음과 같은 해를 정의하자:

$$y_v = \begin{cases} x_v + \epsilon, & v \in V_+ \\ x_v - \epsilon, & v \in V_- \\ x_v, & \text{나머지.} \end{cases},\ z_v = \begin{cases} x_v - \epsilon, & v \in V_+ \\ x_v + \epsilon, & v \in V_- \\ x_v, & \text{나머지.} \end{cases}$$

여기서, $y \neq z$이며 y와 z가 비음 조건과 (12.12)의 가능해가 되도록 ϵ을 충분히 작게 잡을 수 있다.
그런데, $x = \frac{1}{2}(y + z)$. □
이러한 $\frac{1}{2}$-정수성을 사용하면 2-근사해법을 쉽게 만들 수 있다. 왜인지 설명하여 보라.

쌍대성을 사용한 Set Cover 알고리듬

$A = (a_{ij}) \in \mathbb{Q}^{m \times n}$

$$\begin{aligned} \min\ & c^T x \\ Ax &\geq b \\ x &\geq 0. \end{aligned} \tag{13.13}$$

$$\begin{aligned} \max\ & b^T y \\ A^T y &\leq c \\ y &\geq 0. \end{aligned} \tag{13.14}$$

근사-원-상보여유조건

$\alpha \geq 1$
모든 $j = 1, \ldots, n$에 대해, $x_j = 0$이거나,
$\frac{c_j}{\alpha} \leq \sum_{i=1}^{m} a_{ij} y_i \leq c_j$.(또는, $\frac{1}{\alpha} c^T \leq y^T A \leq c^T$.)

근사-쌍대-상보여유조건

$\beta \geq 1$
모든 $i = 1, \ldots, m$에 대해, $y_i = 0$이거나,
$b_i \leq \sum_{j=1}^{n} a_{ij} x_j \leq \beta b_i$.(또는, $b \leq Ax \leq \beta b$.)

기본정리 13.1

(x, y)가 근사-상보여유조건을 만족하는 원-쌍대 가능해 쌍이면 $(b^T y \leq) c^T x \leq \alpha\beta b^T y$이 성립한다.

증명:

$$c^T x \leq \alpha y^T A x \leq \alpha\beta y^T b. \ \square$$

Set Cover를 위한 원-쌍대 알고리듬

$$\alpha = 1, \ \beta = f.$$

근사-원-상보여유조건

$$\forall S, \ x_S > 0 \Rightarrow \sum_{e \in S} y_e = c_S.$$

근사-쌍대-상보여유조건

$$\forall e, \ y_e > 0 \Rightarrow \sum_{S: e \in S} x_S \leq f.$$

알고리듬 13.2

Set Cover 원-쌍대 근사 알고리듬

1. $x \leftarrow 0$, $y \leftarrow 0$;

2. 모든 원소가 커버될 때 까지 다음을 반복한다:
아직 커버되지 않은 임의 원소 e를 선택한다;
등호로 만족되는 쌍대 제약식이 생길 때 까지
y_e를 증가시킨다;
등호로 만족되는 제약식의 집합들을 모두 커버에 넣고
이에 따라 x_S를 업데이트한다;
이 때 새롭게 커버되는 원소들을 업데이트 한다.

3. 집합커버 x를 출력한다.

정리 13.3

알고리듬 13.2은 f-근사해를 보장한다.

Part VIII

Chapter 16

MAX-SAT을 위한 선형계획 라운딩

정의 14.1

Max-Sat

입력: n개의 부울변수 $x = (x_1, x_2, \ldots, x_n)$의 리터럴들의 disjunction으로 이루어진 절(clause) c들의 집합 $\mathcal{C}$의 conjunctive normal form 함수,

$$f = \bigwedge_{c \in \mathcal{C}c} c.$$

각 절 $c \in \mathcal{C}$의 가중치 w_c.

최적해: f의 만족되는 절들의 가중치의 합이 최대가 되는 x의 진리값.

모든 절의 리터럴의 개수가 k개 이하이면 MAX-kSAT이라고 부르자.

절의 리터럴이 많을 때

알고리듬 14.2

Max-Sat 알고리듬

각 변수가 독립적으로 $\frac{1}{2}$ 확률로 True가 되도록 진리값을 생성한다.

해에서 절 c의 가중치 값을 W_c라고 하면, c의 리터럴의 개수를 k라고 하면 $E[W_c] = (1 - \frac{1}{2^k})w_c$가 됨을 쉽게 알수 있다.
$(1 - \frac{1}{2^k}) \geq \frac{1}{2}$이므로 전체 가중치의 기대값은

$$E[W] = \sum_{c \in \mathcal{C}} E[W_c] \geq \frac{1}{2} \sum_{c \in \mathcal{C}} w_c \geq \frac{1}{2}\mathrm{OPT}.$$

모든 절에서 $k \geq 2$이면 $E[W] \geq \frac{3}{4}\mathrm{OPT}$가 됨을 알 수 있다.

Derandomization

알고리듬 14.2를 결정적 알고리듬으로 derandomize하여 보자. 즉, 동일한 근사치를, 확률적이 아니라, 결정적으로 보장할 수 있는 방법을 보자.

$$\begin{aligned} & E[W|x_1 = a_1, \ldots, x_i = a_i] \\ &= E[W|x_1 = a_1, \ldots, x_i = a_i, x_{i+1} = \text{True}] \cdot \tfrac{1}{2} \\ &+ E[W|x_1 = a_1, \ldots, x_i = a_i, x_{i+1} = \text{False}] \cdot \tfrac{1}{2} \end{aligned}$$

따라서, $E[W|x_1 = a_1, \ldots, x_i = a_i, x_{i+1} = \text{True}]$,
또는 $E[W|x_1 = a_1, \ldots, x_i = a_i, x_{i+1} = \text{False}]$는 $E[W|x_1 = a_1, \ldots, x_i = a_i]$보다 커야한다.

따라서 이러한 관계를 x_i들의 임의로 정한 순서에 따라 (첨자 순서라고 가정하자) 적용하면 모든 x_i들이 진리값을 갖게 되는 순간, $E[W]$와 같거나 큰 목적함수를 가진 해가 반드시 따라서 결정적으로 구해지게 된다.

모든 i에 대해 기대값 $E[W|x_1 = a_1, \ldots, x_i = a_i]$를 다항시간에 계산할 수 있다면, 이는 다항시간 알고리듬이 된다. 진리값이 독립적으로 주어지는 경우 이는 다항시간에 계산할 수 있다. (왜 인가?)

Notice: 이 원리는 기대값을 다항시간에 계산할 수 있다면 변수 진리값이 독립적으로 주어지지 않는 경우에도 적용된다:

$$\begin{aligned} & E[W|x_1 = a_1, \ldots, x_i = a_i] \\ = & E[W|x_1 = a_1, \ldots, x_i = a_i, x_{i+1} = \mathrm{True}] \\ & \times \Pr[x_{i+1} = \mathrm{True}|x_1 = a_1, \ldots, x_i = a_i] \\ & + E[W|x_1 = a_1, \ldots, x_i = a_i, x_{i+1} = \mathrm{False}] \\ & \times \Pr[x_{i+1} = \mathrm{False}|x_1 = a_1, \ldots, x_i = a_i]. \end{aligned}$$

절의 리터럴이 적을 때

S_c^+: 절 c에 원형으로 나타나는 변수 집합
S_c^-: 절 c에 부정으로 나타나는 변수 집합

$$y_i = \begin{cases} 1, & x_j = \text{True} \\ 0, & x_j = \text{False}. \end{cases}$$

$$\begin{aligned} & \max \textstyle\sum_{c\in\mathcal{C}} w_c z_c \\ \text{sub. to } & \textstyle\sum_{i\in S_c^+} y_i + \sum_{i\in S_c^-}(1-y_i) \geq z_c \ \forall c \in \mathcal{C} \\ & z_c \in \{0,1\},\ \forall c \in \mathcal{C}, \\ & y_i \in \{0,1\},\ \forall i. \end{aligned} \tag{14.15}$$

$$\begin{aligned} & \max \textstyle\sum_{c\in\mathcal{C}} w_c z_c \\ \text{sub. to } & \textstyle\sum_{i\in S_c^+} y_i + \sum_{i\in S_c^-}(1-y_i) \geq z_c \ \forall c \in \mathcal{C} \\ & 0 \leq z_c \leq 1,\ \forall c \in \mathcal{C}, \\ & 0 \leq y_i \leq 1,\ \forall i. \end{aligned} \tag{14.16}$$

알고리듬 14.3

Max-Sat LP 라운딩 알고리듬
선형계획 14.16를 풀어 최적해 (y^*, z^*)를 구한다.
각 변수 x_i가 확률 y_i^*로 True가 되도록 독립적으로 진리값을 결정한다.

LP 라운딩 알고리듬으로 얻은 진리값이 만족하는 절의 가중치 합을 W, 그 중 절 c에 해당하는 부분을 W_c라고 하자.

기본정리 14.4

c가 k개의 리터럴로 만들어졌다면, $E[W_c] \geq \left(1-\left(1-\frac{1}{k}\right)^k\right) w_c z_c^*$.

증명: c의 변수들을 편의상 $x_1, \ldots, x_k$라고 하자.
그러면 c가 만족될 확률은
$1-\prod_{i \in S_c^+}(1-y_i)\prod_{i \in S_c^-} y_i \geq 1-\left(\frac{1}{k}\left(\sum_{i \in S_c^+}(1-y_i)+\sum_{i \in S_c^-} y_i\right)\right)^k =$
$1-\left(1-\frac{1}{k}\left(\sum_{i \in S_c^+} y_i+\sum_{i \in S_c^-}(1-y_i)\right)\right)^k \geq 1-(1-\frac{z*_c}{k})^k$.
여기서 $g(z) := 1-(1-\frac{z}{k})^k$는 오목(concave)함수이다. 따라서 구간 $[0,1]$에서
$h(z) = g(1)z+g(0) = \left(1-\left(1-\frac{1}{k}\right)^k\right) z$보다 크다. 따라서 $\cdots$.$\square$
$1-\left(1-\frac{1}{k}\right)^k$는 k에 대한 감소함수이다. 따라서, 모든 절의 리터럴의 수가
k를 넘지 않는다면, $E[W] \geq \left(1-\left(1-\frac{1}{k}\right)^k\right)\sum_{c \in \mathcal{C}} w_c z_c^* =$
$\left(1-\left(1-\frac{1}{k}\right)^k\right)\mathrm{OPT}_{LP} \geq \left(1-\left(1-\frac{1}{k}\right)^k\right)\mathrm{OPT} \geq \left(1-\frac{1}{e}\right)\mathrm{OPT}$.
알고리듬 14.2와 마찬가지로 알고리듬 14.3을 derandomize할 수 있다.

$\frac{3}{4}$-근사 알고리듬

다음과 같은 알고리듬을 생각해보자.

동전을 던진다. 앞이 나오면 알고리듬 14.2를 뒤가 나오면 알고리듬 14.3를 수행한다.

기본정리 14.5

$E[W_c] \geq \frac{3}{4} w_c z_c^*$.

증명: $E[W_c|\text{앞}] = (1 - 2^{-k}) w_c \geq (1 - 2^{-k}) w_c z_c^*$,
$E[W_c|\text{뒤}] = \left(1 - \left(1 - \frac{1}{k}\right)^k\right) w_c z_c^*$. 따라서,
$E[W_c] \geq \frac{1}{2} \left((1 - 2^{-k}) + \left(1 - \left(1 - \frac{1}{k}\right)^k\right)\right) w_c z_c^* \geq \frac{3}{4} w_c z_c^*$. □

알고리듬 14.6

(결정적 $\frac{3}{4}$-근사 알고리듬) 1. 알고리듬 14.2을 derandomize한다.
2. 알고리듬 14.3를 derandomize한다.
3. 두 개의 해 중 나은 것을 출력한다.

기본정리 14.7

알고리듬 14.6은 MAX-SAT의 3/4-근사해를 결정적으로 보장한다.

앞에서 $E[W] \geq \frac{3}{4}\mathrm{OPT}_{LP}$, 따라서, 알고리듬이 제공하는 정수해의 integrality gap은 $\frac{3}{4}$ 이상이다. 실제로 다음과 같은 tight example이 존재한다.

예 14.8

$f = (x_1 \vee x_2) \wedge (\bar{x}_1 \vee x_2) \wedge (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_2)$. 모든 절의 가중치를 1이라고 하면 $y_i = \frac{1}{2}$, $z_c = 1$이 선형완화의 최적해가 됨을 알수 있다. 따라서 $\mathrm{OPT}_{LP} = 4$. 그러나, $\mathrm{OPT} = 3$.

예 14.9

$f = (x_1 \vee x_2) \wedge (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee x_2)$. 절의 가중치를 각각 1, 1, $2 + \epsilon$이라고 하면 $y_i = \frac{1}{2}$, $z_c = 1$이 선형완화의 최적해가 됨을 알수 있다. 따라서, 두 randomized 알고리듬 모두 모든 변수를 $\frac{1}{2}$의 확률로 True로 지정한다. derandomization 단계에서 x_1을 가장 먼저 조건으로 잡으면, $E[W|x_1 = \mathrm{True}] = 3 + \frac{\epsilon}{2}$, $E[W|x_1 = \mathrm{False}] = 3 + \epsilon$. 따라서 x_1은 False로 지정한다. 그러면 목적함수는 $3 + \epsilon$. 그러나, x_1를 True로 지정하면 $4 + \epsilon$의 가중치를 얻을 수 있다.

Part IX

Chapter 17

독립 병렬프로세서 스케줄링

문제 15.1

독립 병렬프로세서 스케줄링
입력: 작업 집합 J, 프로세서 집합 M, 요구 작업시간 $p_{ij} \in \mathbb{Z}_+$, $i \in M$, $j \in J$.
최적해: 'makespan' 즉 마지막 작업이 끝나는 순간까지의 시간을 최소화하는 프로세서-작업 스케줄.

p_{ij}가 모든 $i \in M$에 대해 같은 경우, 최소 makespan 문제라고 부르며, PTAS가 존재. 각 프로세서의 속도 s_i가 있어, 작업시간이 p_j/s_i로 결정되는 경우에도 PTAS 존재 (17.5).

Parametric pruning을 사용한 선형완화

$$\begin{aligned} &\min \quad t \\ &\text{sub. to } \sum_{i \in M} x_{ij} = 1, \ \forall j \in J \\ &\sum_{j \in J} p_{ij} x_{ij} \leq t, \ \forall i \in M \\ &x_{ij} \in \{0, 1\}, \ \forall i \in M, \ \forall j \in J. \end{aligned}$$

선형완화문제의 integrality gap은 무한하다:

예 15.2

m개의 프로세서와 모든 프로세서에서 m의 수행시간이 걸리는 한개의 작업의 경우, 최적 목적함수 값은 m이지만 선형완화 최적해 목적함수 값은 1.

이렇게, 목적함수보다 큰 작업시간을 가진 변수가 양의 값을 갖는 극단적인 해를 방지 하기 위해 다음과 같이 parametric pruning을 병행하는 선형계획완화를 생각하자: 각 $T \in \mathbb{Z}_+$값에 대해, $S_T = \{(i,j) : p_{ij} \le T\}$로 정의하자. 그리고 $\mathrm{LP}(T)$를 다음과 같은 선형계획 가능성문제로 정의하자:

$$\begin{aligned} \textstyle\sum_{i:(i,j)\in S_T} x_{ij} &= 1 \;\forall j \in J \qquad (15.17) \\ \textstyle\sum_{j:(i,j)\in S_T} p_{ij}x_{ij} &\le T,\; \forall i \in M \\ x_{ij} &\ge 0,\; \forall (i,j) \in S_T. \end{aligned}$$

이분탐색으로 $\mathrm{LP}(T)$가 가능해를 갖는 최소 T를 구할 수 있다. 이를 T^*라고 하면 OPT의 하한이 된다.

기본정리 15.3

$\mathrm{LP}(T)$의 꼭지점해는 최대 $m+n$개의 양수 원소를 갖는다.

따름정리 15.4

LP(T)의 *꼭지점해는 최소* $n - m$*개의 작업에 대해 정수값 즉* 1*을 갖는다.*

증명: 꼭지점 해에서 정수값을 갖는 작업, 즉 한개의 프로세서에 배치된 작업의 수를 α, 여러 개의 프로세서에 분수값으로 나누어진 작업의 수를 β라고하자. 그러면, 양수를 갖는 원소는 최소한 $\alpha + 2\beta$이지만 앞의 기본정리 15.3에 의하여 양수 원소의 개수는 최대 $m + n$이다. 따라서 $\alpha + 2\beta \leq m + n$. 또한 $\alpha + \beta = n$. 그러므로 $\alpha \geq n - m$. □

LP(T)의 꼭지점해, x에 대해 다음과 같은 마디집합 $J \cup M$를 갖는 이분그래프 $G = (J \cup M, E)$를 정의한다. $(j, i) \in E \Leftrightarrow x_{ij} \neq 0$. $F \subseteq J$를 여러 개의 프로세서에 분수값으로 나누어진 작업의 집합이라고 하자. 그리고 H를 마디 집합 $F \cup M$에 의하여 유도된 G의 부분그래프라고 하자. 물론 H의 각 호 (j, i)는 $0 < x_{ij} < 1$를 만족한다.

H에는 F의 마디들을 모두 커버하는 짝짓기가 존재한다는 것을 증명할 수 있다(기본정리 15.6).

<u>꼭지점 해의 성질</u>

마디집합 V를 가지며, 호의 개수가 $|V|$ 이하인 연결된 그래프를 *유사나무*(pseudo-tree)라고 부르자. 또한 연결요소들이 모두 유사나무인 그래프를 *유사숲*(pseudo-forest)이라고 하자.

기본정리 15.5

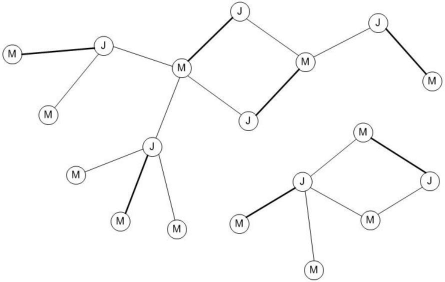
LP(T)의 꼭지점해로 정의된 이분그래프 $G = (J \cup M, E)$는 유사숲이 된다.

증명: G의 임의의 연결 요소를 G_c라고 하자. LP(T)를 G_c의 작업과 프로세서에 국한하여 얻은 문제를 LP$_c$(T)라고 하자. 그리고 x중 G_c의 작업과 프로세서에 대응되는 부분을 x_c라고 하자. 나머지는 $x_{\bar{c}}$라고 하자. G_c가 G의 다른 부분과 연결되지 않은 요소이기 때문에 x_c는 LP$_c$(T)의 꼭지점이 된다. (만약 아니라면, x_c는 LP$_c$(T)의 서로 다른 두 점의 볼록조합으로 표현되는데, G_c가 독립된 연결요소이기 때문에 각 점은 $x_{\bar{c}}$를 합하면 LP(T)의 가능해가 된다. 이렇게 얻은 두 가능해를 볼록조합하면 x가 되며 이는 x가 꼭지점이라는 것에 모순이 된다.) 따라서, 기본정리 15.3를 적용하면 G_c는 유사나무가 된다. □

기본정리 15.6

H는 모든 작업을 커버하는 짝짓기를 갖는다.

증명: x에서, 오직 한 개의 프로세서에 1로 할당되는 작업들과 그 1의 값에 대응되는 호들을 제거하여 얻은 그래프가 바로 H가 된다. 즉 H는 G에서 같은 갯수의 마디와 호가 제거되어 얻은 그래프이므로 역시 유사숲이 된다. 또한 H는 각 작업이 최소 2개의 호를 달고 있다(아래 그림 참조).



따라서 H의 모든 잎(leaf)은 프로세서가 된다. 각 잎을 이웃한 작업과 짝짓고 이 두 마디를 H에서 제거한다. 이를 모든 연결요소에 반복 적용한다. 이 때, 매 단계 마다 모든 잎은 프로세서가 된다. (왜인가?)

만약 이런 과정에서 그래프의 어떤 연결요소의 부분이 남는다면 이는 회로가 되어야 하며, 이분그래프이기 때문에 작업과 프로세서가 번갈아 나타나는 짝수 회로가 된다. 따라서, 작업과 프로세서를 남김없이 짝지울 수 있다. □

알고리듬

우선 T의 범위를 구해보자. 각 작업을 최소 수행 시간을 갖는 프로세서에 대응시키는 해를 생각해 보자. 어떠한 선형완화 해의 총 작업시간의 합도 이 해의 경우보다 작을 수 없다. 한편 이 해의 makespan을 α라고 하면, 이 해는 pruning을 병행한 선형완화문제 $\mathrm{LP}(\alpha)$의 가능해이다. 따라서, T^*는 구간 $[\alpha/m, \alpha]$의 범의 안에 있다.

알고리듬 15.7

1. 구간 $[\alpha/m, \alpha]$의 이분탐색을 통하여 $\mathrm{LP}(T)$가 가능해를 갖는 최소 T를 구한다. 이를 T^*라고 하자;
2. $\mathrm{LP}(T^*)$의 꼭지점해 x를 구한다;
3. 정수값을 갖는 변수들은 대응하는 작업들과 프로세서들을 짝지운다;
4. 앞에서 설명한 것과 같이 그래프 H를 구성하고 모든 작업을 커버하는 짝짓기를 찾는다:
5. F의 작업들을 이 짝짓기에 따라 배정한다.

정리 15.8

알고리듬 15.7은 2-근사를 보장한다.

증명: $T^* \leq \mathrm{OPT}$임을 쉽게 알 수 있다. 꼭지점해 x가 makespan T^*를 갖고 있기 때문에, 단계 3까지 모든 프로세서는 T^*안에 작업을 끝내게 된다. 또한 H의 호들은 모두 작업시간이 T^*를 넘지 않는다. 그런데 단계 5에서 각 프로세서마다 최대 한개의 작업을 추가 배정 받으므로, 최종 해의 makespan은 $2T^*$를 넘지 않는다. □

Tight example

m개의 프로세서와 $1 + m(m-1)$개의 작업을 생각하자. 처음 한 개의 작업은 모든 프로세서에서 m의 시간이 필요하다. 나머지 작업은, 모두, 어떤 프로세서에서든지 단위시간이 필요하다고 하자. 이 경우, 최적해는 $\mathrm{OPT} = m$.

알고리듬 15.7를 적용하여 보자. $T < m$이면 $\mathrm{LP}(T)$는 가능해가 존재하지 않는다. 따라서 $T^* = m$. $\mathrm{LP}(T)$는 다음과 같은 꼭지점해를 갖는다. 처음 한 개의 작업은 모든 프로세서에 $\frac{1}{m}$만큼 할당되고, 나머지 작업은 모두 $m-1$개씩 m개의 프로세서에 1로 할당. (꼭지점임을 확인 할 것. 연습문제 17.2)
이 때, 알고리듬 15.7이 생성한 해의 makespan은 $2m-1$이 된다.

Part X

Chapter 18

Multicut과 정수다품목흐름문제: 나무의 경우

정의 16.1

multicut
입력: 무향그래프 $G = (V, E)$, 호 용량 $c_e \in \mathbb{Q}_+$, $e \in E$, k개의 마디 쌍 집합, $\{(s_1, t_1), \ldots, (s_k, t_k)\}$.
최적해: 제거되면 모든 쌍 (s_i, t_i)가 서로 분리되는 호 집합 중 최소 용량을 가진 것.

최소비용 터미널 절단면 문제는 multicut문제로 쉽게 변환된다: s_1, s_2, s_3를 모두 서로 분리되도록 하는 것은, 다음과 같은 마디쌍들이 모두 분리하는 것과 같다: (s_1, s_2), (s_2, s_3), (s_3, s_1). 따라서, NP-hard문제이다.

multicut은 $k \geq 2$이면 NP-hard이다. $k \geq 2$이면 max-SNP-hard이며 따라서 어떤 c-근사해법은 불가능한 상수 c가 존재한다. 일반적인 경우, 현재 $O(\log k)$의 근사계수를 갖는 알고리듬이 존재한다. G가 나무인 경우에도 max-SNP-hard. 그러나 이 경우 2-근사해법이 가능하다.

성질 16.2

G가 높이 1이고 단위 용량의 호를 가진 나무인 경우에도 multicut문제는 NP-hard이다.

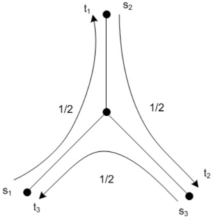
증명: 마디커버문제를 위의 성질을 가진 multicut문제로 변환할 수 있다.□

나무 multicut 문제를 위한 정수계획모형

$$\begin{aligned} &\min \textstyle\sum_{e\in E} c_e d_e \\ \text{sub. to } &\textstyle\sum_{e\in P_i} d_e \geq 1,\ i = 1, \ldots, k \\ &d_e \in \{0, 1\}. \end{aligned}$$

$$\begin{aligned} &\min \textstyle\sum_{e\in E} c_e d_e \\ \text{sub. to } &\textstyle\sum_{e\in P_i} d_e \geq 1,\ i = 1, \ldots, k \\ &d_e \geq 0. \end{aligned}$$

두 문제 사이에는 다음의 예에서 볼 수 있는 것처럼 정수간격이 존재한다.



선형완화문제의 쌍대문제는 다음과 같다. 연속적인 값을 갖는 다품목흐름문제로 해석할 수 있다. 이 때, 각 호에서, 방향에 관계 없이, 모든 품목흐름의 흐름의 합은 용량 조건을 만족하여야 한다.

연속다품목흐름문제

$$\begin{aligned} &\max \textstyle\sum_{i=1}^{k} f_i \qquad (16.18)\\ \text{sub. to } &\textstyle\sum_{i: e \in P_i} f_i \leq c_e, \;\; e \in E \\ &f_i \geq 0 \;\; i \in \{1, \ldots, k\}. \end{aligned}$$

문제 16.3

정수다품목흐름문제
입력: multicut문제와 동일. 각 (s_i, t_i)를 품목 i의 'source'와 'sink'로 생각한다. 단, 용량은 모두 정수.
출력: 모든 품목 흐름 크기의 합이 최대가 되는 정수 흐름.

정수다품목흐름문제는 임의의 용량을 가지면, 높이가 3인 나무의 경우도 NP-hard가 된다. 연속 그리고 정수다품종흐름문제 사이에는 정수 간격 (integrality gap)이 존재하는 것을 위의 나무의 경우에서도 알 수 있다.

원-쌍대 근사해법

다음과 같은 근사상보여유성을 추구한다.

원상보성: $d_e > 0 \;\Rightarrow\; \sum_{i:e\in P_i} f_i = c_e$. 즉, multicut에 포함되는 호의 흐름은 꽉 차야 한다.

근사쌍대상보성: $f_i > 0 \;\Rightarrow\; (1 \leq) \sum_{e\in P_i} d_e \leq 2$.
즉, 흐름을 갖는 (s_i, t_i)-경로 중에서 multicut에 포함되는 호는 두 개를 넘지 않는다.

G의 임의의 마디를 뿌리로 지정한다. 각 마디로부터 뿌리까지 경로의 길이를 마디의 깊이로 정의한다. 두 마디 $u, v \in V$를 잇는 경로 중에서 가장 얕은 깊이의 마디를 $\mathrm{lca}(u, v)$로 표기하자.

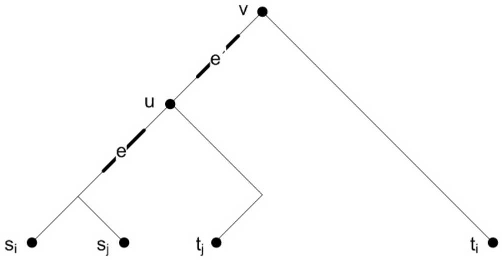
알고리듬 16.4

1. $f \leftarrow 0$, $D \leftarrow \emptyset$;
2. 깊이가 낮은 순으로 각 마디 v에 대하여 다음을 반복한다: $\mathrm{lca}(s_i, t_i) = v$인 모든 (s_i, t_i)에 대하여 흐름을 최대 가능한 만큼 배정한다. 이 때, 포화된 호들을 D에 순서대로 첨가한다. 같은 반복단계에 첨가된 호들은 순서를 임의의 정한다.
3. D에 첨가된 호들을 순서대로 $e_1, e_2, \ldots, e_l$이라고 하자.
4. 첨가된 역순으로 각 호 e에 대하여 다음을 수행한다: 만일 $D \setminus \{e\}$가 multicut으로 유지되면 $D \leftarrow D \setminus \{e\}$로 수정한다.
5. f와 D를 출력한다.

기본정리 16.5

(s_i, t_i)를 0보다 큰 흐름을 갖는 쌍, 그리고 $\mathrm{lca}(s_i, t_i) = v$라고 하자. 그러면 s_i에서 v, 그리고 v에서 t_i까지의 경로 각각에서 최대 한 개 호가 D에 포함된다.

증명: s_i에서 v까지 경로의 두 개의 호 e와 e'이 D에 포함되었다고 하자. 그리고 e가 e'보다 깊이 있다고 하자. 위의 알고리듬 단계 4에서 e를 고려하는 순간을 생각해보자. e가 제거되지 않았으므로, D에서 e를 유일하게 그 사이 경로에 갖는 마디쌍 (s_j, t_j)가 존재한다. 이 때, e'은 $\mathrm{lca}(s_j, t_j)$보다 위에 있어야 하기 때문에, $u := \mathrm{lca}(s_j, t_j)$는 $v = \mathrm{lca}(s_i, t_i)$ 보다 깊이 존재한다.



u가 단계 2에서 고려된 후에 D는 (s_j, t_j)를 연결하는 경로 위의 어떤 호 e''을 D에 포함시킬 것이다. 가정에서 v를 고려할 때, (s_i, t_i)를 연결하는 경로 위에 0보다 큰 흐름을 보낼 수 있었다는 것은, e는 그 전까지는 포화상태에 있지 않았다는 것을 의미하므로, v를 단계 2에서 고려한 후에 D에 포함되었는 것을 의미 한다. u는 v보다 더 깊은 위치에 있으므로 먼저 고려되며, 따라서 e''은 e보다 먼저 D에 포함 되었기 때문에, 단계 4에서 e가 고려될 때, e''은 D에 존재한다. 이는 e가 (s_j, t_j) 경로 위의 유일한 D의 호라는 것에 모순. □

정리 16.6

알고리듬 16.4는 multicut문제의 2-근사를, 정수다품목문제에는 $\frac{1}{2}$-근사를 보장한다.

증명: 알고리듬의 해가 각각 multicut과 정수다품목흐름문제의 가능해가 된다는 것은 쉽게 알 수 있다. 또한 각 경로에서 흐름이 포화되는 호들을 D에 넣었기 때문에 원상보성이 성립한다. 또한 기본정리 16.5에 의하여 근사쌍대상보성도 성립하는 것을 알 수 있다. 따라서 기본정리 13.1에서 $\alpha = 1$, $\beta = 2$가 만족하므로 $c(D) \leq 2|f|$이 성립. $|f|$는 최적 multicut의 하한이고, $c(D)$는 최적 정수다품목흐름문제의 상한이되므로 정리가 성립한다. □

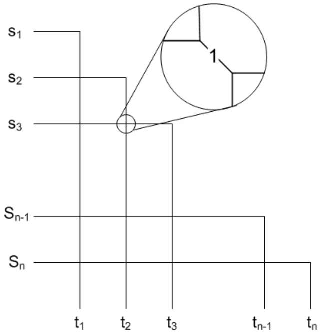
이로부터 다음과 같은 근사적인 min-max 관계를 얻는다.

따름정리 16.7

호의 용량이 정수인 나무에서는 다음과 같은 근사적인 최대 정수다품목흐름 최소 multicut 용량 관계가 성립한다:

$$\max_{\text{정수흐름 } f} |f| \leq \min_{\text{multicut } C} c(C) \leq 2 \max_{\text{정수흐름 } f} |f|.$$

일반적인 그래프에서는 $O(\log k)$-근사해법이 가능한데 이 경우, 하한을 분수 multicut을 사용하게 된다. 그러나, 정수다품종흐름 문제의 경우, 나무보다 일반적인 그래프에선 아직 어떠한 근사해법도 알려져 있지 않다. 실제로 다음의 예는 평면(planar) 그래프에서도 다품목흐름문제의 선형완화문제의 정수간격이 최소 $\frac{n}{2}$라는 것을 보여준다.


s1
s2
s3
Sn-1
Sn
1
t1
t2
t3
tn-1
tn

Part XI

Chapter 20

일반 그래프의 multicut 문제

최대흐름-최소절단면 정리(max-flow min-cut theorem)의 다품목 일반화.

1. 최대 다품목 흐름합 문제: 동시에 흐르는 품목들의 흐름의 합을 최대화 - multicut문제와 연관.
2. 최대 다품목 흐름율 문제: 각 품목 i의 수요 d_i를 미리 정해 놓고 모든 품목이 $f \cdot d_i$만큼 동시 흐를 수 있는 흐름율(throughput) f를 최대화- sparsest cut문제와 연관

<u>최대 다품목 흐름합 문제</u>

문제 17.1

입력: 무향 그래프 $G = (V, E)$, 호 용량 $c_e \in \mathbb{Q}_+$, 서로 다른 마디쌍 $\{(s_1, t_1), \ldots, (s_k, t_k)\}$ (각 쌍은 서로 다른 품목의 source와 sink).
최적해: G에서 모든 품목들의 흐름크기 합을 최대화. 이 때, 각 품목의 흐름보존과 각 호의 용량 제한을 모두 만족해야한다. 이 때, 같은 방향뿐 아니라 반대 방향으로 흐르는 품목들의 흐름 크기의 합이 호 용량을 넘어서는 안된다.

최각 품목 i의 모든 (s_i, t_i)-경로 집합을 $\mathcal{P}_i$, 그리고 $\mathcal{P} = \bigcup_{i=1}^k \mathcal{P}_i$라고 하자. $\mathcal{P}$의 각 경로 P에 호르는 흐름의 크기를 f_P라고 하자.

$$\max \textstyle\sum_{p\in\mathcal{P}} f_P \tag{17.19}$$
$$\text{s.t. } \textstyle\sum_{P:e\in P} f_P \le c_e,\ e \in E \tag{17.20}$$
$$f_P \ge 0,\ P \in \mathcal{P} \tag{17.21}$$

쌍대문제 변수를 d_e, $e \in E$라고 하자. 이때 d_e를 호 e의 거리로 해석하자.

$$\min \textstyle\sum_{e\in E} c_e d_e \tag{17.22}$$
$$\text{s.t. } \textstyle\sum_{e\in P} d_e \ge 1,\ P \in \mathcal{P}$$
$$d_e \ge 0,\ e \in E$$

쌍대해 d가 가능해가 될 필요 조건은, 이 거리 값을 정의된 (s_i, t_i) 최단경로가 최소 1,이 되는 것이다. (따라서 분리문제를 최단경로문제로 다항시간에 풀 수 있으므로 쌍대문제는 다항시간에 풀 수 있다.) 이 쌍대문제의 정수 최적해는 바로 최소비용 multicut이 된다.

최대 다품목 흐름합 문제의 최적해의 $O(\log k)$ 배가 되는 정수해를 구하는 알고리듬을 공부하게 되는데, 이는 nulticut과 최대 다품목 흐름합 문제의 쌍대 간격이 $O(\log k)$를 넘지 않는 것을 의미한다.

LP-라운딩 알고리듬

최소비용 multicut의 LP-완화문제 (17.22)의 최적해 d_e의 목적함수 값 $F = \sum_{e \in E} c_e d_e$를 생각하자. 자연스럽게 F를 하한으로, 이에 비해 크지 않은 비용을 갖는 multicut D를 찾는 근사해법을 생각할 수 있다.

(예를 들어 거리 값이 양수인 호를 모두 포함하면 multicut은 되지만 F에 비해 매우 큰 비용을 가질 수 있다. (연습문제 20.3, 20.4))

이를 위해 multicut 그래프 $G = (V, E)$에서 각 호 e의 거리를 d_e로, 가중치를 $c_e d_e$로 정의하자. 또한 현재 거리 값 d_e $(e \in E)$에 대하여 마디 u와 v간의 최단 경로의 길이를 $d(u, v)$로 표기하자.

우리의 근사해법은 다음의 두 조건을 만족하는, 서로 소인 마디 집합들, S_1, S_2, ..., S_l $l \leq k$를 찾는다. 각 집합을 영역이라고 부르자.

- 어떤 i도 s_i와 t_i가 모두 같은 영역에 포함되지 않는다. 모든 i는, s_i와 t_i 중 최소 하나가 어떤 영역에 포함된다.
- 각 영역 S_i에 의하여 정의된 절단면 $\delta(S_i)$의 절단면의 용량 $c(S_i)$는, 어떤 $\epsilon > 0$에 대해 $c(S_i) \leq \epsilon w(S_i)$을 만족한다.

여기서, $w(S)$는 영역 S의 가중치로, 우선 대략적으로 말하면 S에 양 끝 노드를 갖는 호의 가중치의 합으로 정의된다.
첫 번째 조건은 $M = \delta(S_1) \cup \delta(S_2) \cup \cdots \cup \delta(S_l)$이 multicut임을 보증한다. 두 번째 조건은, 대략적으로 말해, multicut M의 용량이 $O(\epsilon)F$(정도)가 됨을 말한다.

영역 확장 과정

영역 S_1, S_2, ..., S_l $l \leq k$는 다음과 같은 확장 과정을 거쳐 구성된다. 각 영역은 어떤 i에 대해 s_i나 t_i 중 하나의 원소로 시작한다. 이를 영역의 뿌리라고 한다. 뿌리가, 예를 들어, s_1이라고 하자. 이 뿌리를 중심으로 영역의 반지름을 증가시킨다. 즉, 각 $r \geq 0$에 대해, $S(r)$을 뿌리로부터의 거리가 r을 넘지 않는 마디들의 집합이라고 하자: $S(r) = \{v : d(s_1, v) \leq r\}$. $S(0) = \{s_1\}$로 시작하여 r을 연속적으로 증가시키면, $S(r)$은, s_1으로부터 거리가 작은 순으로 마디가 첨가되어 (불연속적으로) 확장하게 된다.

기본정리 17.2

r이 $\frac{1}{2}$이 되기 전에 영역 확장 과정이 종료되면 $S(r)$은 어떠한 i에 대해서도 s_i와 t_i모두 포함할 수 없다.

증명: $S(r)$의 모든 마디 간의 거리는 $2r$을 넘을 수 없다. 제약조건에서 모든 i에 대해 $d(s_i, t_i) \geq 1$이다.□

<u>가중치 $w(S)$</u>

이제 마디 집합 S의 가중치 $w(S)$를 정확히 정의하자. 우선 $w(\{s_1\}) = F/k$로 놓는다. S에 한 끝 마디라도 포함되는 호 e들은 그 포함되는 비율 q_e 만큼 가중치를 S에 포함시킨다. 즉, 만약 e의 양끝 마디가 모두 S에 포함되면 $q_e = 1$로 정의한다. 만약 $e = (u, v)$이고 $u \in S(r)$, $v \notin S(r)$이면,

$$q_e = \frac{r - d(s_1, u)}{d(s_1, v) - d(s_1, u)}.$$

그리고 이에 따라, S의 가중치는 다음과 같이 정의한다:

$$w(S(r)) = w(\{s_1\}) + \sum c_e d_e q_e.$$

제시하는 알고리듬은 $r < \frac{1}{2}$ 범위에서 $c(S(r)) \leq \epsilon w(S(r))$을 만족하는 ϵ과 r을 추구한다.

기본정리 17.3

$\epsilon = 2\ln(k+1)$으로 놓으면 어떤 $r < \frac{1}{2}$에 대하여 $c(S(r)) \leq \epsilon w(S(r))$이 성립하게 된다.

증명: 모든 $r \in [0, \frac{1}{2}]$에서 $c(S(r)) > \epsilon w(S(r))$이 성립한다고 하자. 모든 점에서 $\mathrm{d}w(S(r))/\mathrm{d}r = \sum_e c_e d_e\,(\mathrm{d}q_e/\mathrm{d}r)$이 성립한다. (여기서 우변에서는, 물론, $S(r)$에 한 쪽 끝만을 가진 호들만을 합한다.) $e = (u, v)$를 그러한 호 중에 하나라고 하자: $u \in S(r)$, $v \notin S(r)$. 그러면,

$$c_e d_e \mathrm{d}q_e = \frac{c_e d_e}{d(s_1, v) - d(s_1, u)}\mathrm{d}r.$$

그런데, $d_e \geq d(s_1, v) - d(s_1, u)$이므로 $c_e d_e \mathrm{d}q_e \geq c_e \mathrm{d}r$이 성립하며 따라서,

$$\mathrm{d}w(S(r)) \geq c(S(r))\mathrm{d}r > \epsilon w(S(r))\mathrm{d}r. \tag{17.23}$$

$w(S(0)) = \frac{F}{k}$, $w(S(\frac{1}{2})) \leq F + \frac{F}{k}$. 그리고,

$$\int_{f(a)}^{f(b)} g(f)\mathrm{d}f = \int_a^b g(f(x))f'(x)\mathrm{d}x.$$

이를 (17.23)와 결합하면

$$\int_{\frac{F}{k}}^{F+\frac{F}{k}} \frac{1}{w}\mathrm{d}w > \int_0^{\frac{1}{2}} \frac{1}{w(S(r))}\epsilon w(S(r))\mathrm{d}r = \int_0^{\frac{1}{2}} \epsilon \mathrm{d}r.$$

그러므로, $\ln(k+1) > \frac{1}{2}\epsilon$. 이것은 가정에 모순. □

알고리듬

알고리듬의 효율성을 위하여, 영역을 확장하는 과정은 이산적인 과정으로 수정된다. $S = \{s_1\}$으로 시작하여, s_1로 부터 거리가 작은 마디로부터 차례로 S에 첨가한다. S의 가중치 $w(S)$는 다음과 같이 수정된다:

$$w(S(r)) = w(\{s_1\}) + \sum c_e d_e.$$

단, 우변의 합에서는 S에 최소한 한 끝을 갖는 호를 모두 포함하며, $w(\{s_1\}) = \frac{F}{k}$로 정의한다. 이 과정은 $c(S) \le \epsilon w(S)$이 처음 만족되는 순간 종료된다. 정의에 따라 같은 S에 대해 연속적인 과정의 가중치보다 이산적 과정의 가중치가 같거나 크다. 따라서 이산적 과정에서 생성되는 S는 연속적 과정보다 클 수 없으며, 따라서 어떤 i에 대해서도 s_i와 t_i를 모두 포함할 수 없다.

이러한 이산적 확장 과정을 사용하는 알고리듬은 다음과 같다. 단, 아래 첨자 H는 부분그래프에서 정의된 각 값을 의미한다.

알고리듬 17.4

최소 multicut 근사해법
1. 선형완화문제 (17.22)을 풀어 G의 각 호의 거리 값을 구한다;
2. $\epsilon \leftarrow 2\ln(k+1)$, $H \leftarrow G$, $M \leftarrow \emptyset$;
3. H에 s_i와 t_i가 모두 포함되는 어떤 i가 존재할 때까지 다음을 반복한다:

 임의의 s_j를 선택한다;

 $c_H(S) \leq \epsilon w_H(S)$가 만족될 때까지 영역 S를 확장한다;

 $M \leftarrow M \cup \delta_H(S)$, $H \leftarrow H \setminus S$;

4. M을 출력;

기본정리 17.5

알고리듬 17.4이 출력하는 M은 multicut 이다.

증명: 어떤 영역도 source-sink 쌍을 포함하지 않는다는 것을 보이면 된다. 이를 위해, 기본정리 17.2과 17.3가 모든 반복단계에서 성립함을 보이면 된다. 임의의 반복단계의 부분그래프 H를 생각하자. 우선 H에서의 거리는 G에서는 거리보다 같거나 길어지기 때문에 기본정리 17.2가 성립하는 것은 쉽게 알 수 있다. 또한, H에서 확장되는 영역 S는 하한 $\frac{F}{k}$와 상한 $\frac{F}{k}+F$를 갖는다. 따라서 기본정리 17.3의 증명은 H에서도 그대로 성립한다. □

기본정리 17.6

$c(M) \leq 2\epsilon F = 4\ln(k+1)F.$

증명: i번째 반복단계의 H와 S를 각각 G_i와 S_i로 표기하자. $(G = G_1.)$ 그러면 앞의 증명에서도 언급한 것과 같이, $c_{G_i}(S_i) \leq \epsilon w_{G_i}(S_i)$가 모든 반복단계에서 성립한다. 또한 $w_{G_i}(S_i)$의 호들은 해당 i번째 반복단계가 종료하면 모두 제거된다. 각 반복단계마다 최소 한쌍의 source-sink 쌍이 분리되기 때문에, 수행되는 반복단계의 횟수는 k를 넘지 않는다. 따라서,

$$c(M) = \textstyle\sum_i c_{G_i}(S_i) \leq \epsilon\left(\sum_i w_{G_i}(S_i)\right)$$
$$\leq \epsilon\left(k\frac{F}{k} + \textstyle\sum_e c_e d_e\right) = 2\epsilon F. \square$$

정리 17.7

알고리듬 17.4는 $O(\log k)$-근사해를 보장한다.

따름정리 17.8

k-개의 source-sink 쌍을 가진 최대 다품목 흐름합 문제는 다음과 같은 근사적인 최대다품목흐름-최소multicut용량 관계를 갖는다:

$$\max_{\text{흐름 } f} |f| \leq \min_{\text{multicut } C} c(C) \leq O(\log k) \max_{\text{흐름 } f} |f|.$$

<u>Tight example</u>

예 17.9

선형완화문제 (17.22)가 $\Omega(\log k)$의 정수간격을 갖는 예를 만들 수 있다. 이는 알고리듬의 분석이나 위에 기술된 최대다품목흐름-최소multicut용량 관계가 실제로 상수배차이로 tight하다는 것을 의미한다. "Expander" 그래프.

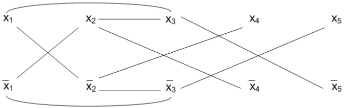
multicut의 응용

예 17.10

최소비용 2CNF≡절 제거문제.
2CNF≡ formula F는 u와 v가 각각 리터럴일 때, $(u \equiv v)$의 형태의 절들의 conjunction을 말한다. 각 절에 가중치 w가 주어졌을 때, 제거되면 F가 만족되는 절들의 부분집합 중에서 가중치의 합이 최소가 되는 것을 찾는 문제를 생각하자. F가 n개의 변수로 이루어졌다고 하자.

다음과 같이, 각 변수의 두 개의 리터럴에 각각 한 개의 마디를 대응시켜, 모두 $2n$개의 마디를 갖는 그래프 $G(F)$를 생각하자. 각 절 $(p \equiv q)$에 두 개의 호 (p, q), $(\bar{p}, \bar{q})$를 대응시킨다. 두 개의 호 모두, 절 $(p \equiv q)$의 가중치와 같은 용량을 할당한다. 동등한 절 $(p \equiv q)$와 $(\bar{p} \equiv \bar{q})$가 둘 다 존재하면 합쳐서 하나로 대체할 수 있다. 따라서, F의 각 절에 $G(F)$의 두 개의 호가 대응된다고 가정할 수 있다.

$$(x_1 \equiv \bar{x}_2) \wedge (x_1 \equiv x_3) \wedge (x_2 \equiv x_3) \wedge (\bar{x}_2 \equiv x_4) \wedge (x_3 \equiv \bar{x}_5).$$



기본정리 17.11

F가 만족될 필요충분조건은 $G(F)$의 어떤 연결 요소도 한 변수와 그의 부정을 포함하지 않는 것이다.

증명: 충분 조건을 증명하여 보자. 리터럴 p와 q가 한 개의 요소에 나타나면 반드시 그 부정형들도 한 개의 요소에 나타난다. 따라서, 어떠한 요소도 한 변수와 그 부정형을 포함하지 않는다면, 모든 요소들을, 어떤 리터럴 집합과 그 부정형으로 이루어진 두 개의 요소끼리 짝 지울 수 있다. 이런 한 쌍에 속하는 두 절의 부정형으로 대응하는 리터럴들은 진리값을 반대로 갖게 하면 모든 절을 만족하는 진리값을 구하게 된다. □

(위에서 한개의 절에 두개의 호를 모두 대응시키는 것이 왜 피할 수 없는 것인지를 스스로 확신할 것.)

최소비용 2CNF≡절 제거문제에 다음과 같은 $G(F)$의 최소비용 multicut문제를 대응시켜보자: 각 변수에 대응되는 두 개의 마디들을 모두 source-sink 쌍으로 지정하자. M을 이 multicut문제의 최적해라고 하고, C를 2CNF≡절 제거문제의 최적해라고 하자. (M은 일반적으로 각 절에 대응되는 두 개의 호중에서 한 개만을 포함할 수 있다는 것에 유의.) 그러면

기본정리 17.12

$w(C) \leq c(M) \leq 2w(C)$.

증명: M을 제거하면 분명 모든 변수와 그 부정형은 같은 요소에 있지 않게 된다. 기본정리 17.11에 따라, F는 만족되며 첫번째 부등식이 증명된다. 두 번째 부등식은 C의 절에 두 개씩 대응하는 $G(F)$의 호집합이 multicut이며 그 가중치 합이 $2w(C)$라는 것에서 알 수 있다. □

Part XII

Chapter 26

최대절단면문제를 위한 0.878-근사해법

Goemans와 Williamson의 최대절단면문제 근사해법연구는, 다른 접근법으로 달성할 수 있는 최대절단면문제의 최대 가능 근사값(approximation ratio)을, SDP를 사용하면 획기적으로 개선할 수 있다는 것을 보인 최초의 연구이다. 제안된 해법은 확률알고리듬이며, 앞의 분류에서 무작위초평면 기법에 해당한다.
이 연구는, 미국 애틀란타 죠지아 공대에서 열린, 2000년 수리계획학회 (mathematical programming society)에서 풀커슨상을 수상하였다.

최대절단면문제

주어진 무방향의 그래프를 $G = (V, E)$라고 하고, 호의 가중치를 $w_{ij}, (i, j) \in E$로 표시하기로 하자. G의 절단면(cut)이란, 노드집합(node set)을 양분했을 때, 두 노드집합에 걸치는 호(edge)들의 집합을 말한다.

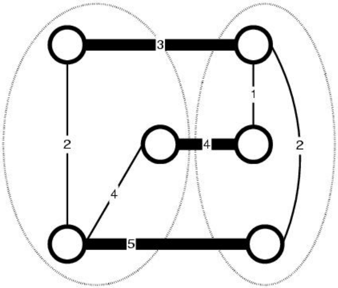


Figure: 최대절단면(cut)의 예

최대절단면문제는 호의 가중치의 합이 최대가 되는 절단면을 구하는 문제이다([그림.1]참조). 즉,

$$w(S) \equiv \sum_{i \in S,\ j \in \bar{S}} w_{ij} \tag{18.24}$$

를 최대화하는 노드의 부분집합 $S \subset V$를 구하는 문제이다.

최대절단면문제는 다항시간 안에 정확한 해를 구하는 알고리즘은 존재하지 않는 것으로 생각되는 NP-hard 문제이다. 특히 다항식근사해법군이 불가능한 max-NP-hard 문제이다.

정의 18.1

어떤 최소화 (최대화) 문제 P의 모든 예(instance)에 대해, 최적목적함수값의 δ 배를 넘지 않는 (넘는) 해를 보장하는 다항시간 해법 $\mathcal{A}$를 *δ-근사해법* (δ-approximation algorithm) 이라고 부른다.

연습문제 18.2

최대절단면문제의 아주 간단한 $\frac{1}{2}$-근사해법: 각 노드 별로 동전을 던진다. 앞이 나오면 V_1에 뒤가 나오면 V_2에 속하도록 노드집합을 무작위로 양분한다: $V = V_1 \dot{\cup} V_2$.

놀라운 사실은 다음과 같은 근사해법이 개발되기 전 약 20년 동안, 위의 간단한 근사해법 보다, 엄밀한 의미에서 더 나은 해법이 존재하지 않았다는 것이다.

물론 모든 *NP*-hard 조합최적화문제의 근사값을 정확한 최적해에 해당하는 1에 무한히 가깝게 개선할 수 있는 것은 아니다. 이러한 조합최적화문제는 소수에 해당하며, 많은 문제들이 알고리듬의 수행시간을 지수적으로(exponentially) 늘이지 않는한, 어떤 값보다 좋은 해를 보장하는 것 자체가 *NP*-hard인 경우가 많다. 이러한 값을 불가능 근사값(impossible approximation ratio)이라고 부른다.

현재까지 알려진 최대절단면문제의 불가능 근사값은 0.94이다. 즉 최적해의 94%의 목적함수 값을 갖는 해를 보장하는 것은 불가능하다는 것이다. Goemans와 Williamson의 무작위 초평면해법은 약 88%의 기대값을 보장하며, 그전의 50%에 비해 불가능 근사값에 획기적으로 가까워진 것을 알 수 있다.

SDP완화를 이용한 근사해법

최대절단면문제의 2차함수모형

최대절단면문제가 $0-1$ 2차함수계획법과 같은 문제임은 잘 알려진 사실이다. Goemans와 Williamson은 다음과 같은 $-1,+1$ 2차함수계획법 모형에서 출발하였다. $V = V_1 \dot\cup V_2$로 양분할 때, $i \in V_1$이면 $x_i = -1$, $i \in V_2$이면 $x_i = 1$로 $x \in \mathbb{R}^{|V|}$를 정의하자. 그러면 최대절단면문제는 다음의 $-1,+1$ 2차함수계획법문제와 같다.

문제 18.3

비대칭형

$$\begin{array}{ll} \max & \frac{1}{2}\sum_{i<j} w_{ij}(1 - x_i x_j) \\ \text{s.t.} & x_i \in \{-1,\ 1\}\ \forall i \in V \end{array}$$

앞장의 비대칭형은 $w_{ij} = w_{ji}$을 가정하면 다음과 같다:

문제 18.4

대칭형

$$\begin{array}{ll} \max & \frac{1}{4}\sum_{i,j} w_{ij}(1 - x_i x_j) \\ \text{s.t.} & x_i \in \{-1,\ 1\}\ \forall i \in V \end{array}$$

SDP완화를 이용한 근사해법
벡터완화

앞절의 $-1, +1$ 2차함수계획법 모형은 $0-1$ 2차함수계획법의 약간의 변형일 뿐, 그 자체로 보면 해법에 있어서 아무런 특별한 장점이 없다. Goemans와 Williamson의 근사해법의 첫번째 도약은 이 $-1, +1$ 2차함수계획법 모형을 다음과 같이 '벡터계획법'문제로 완화했다는데 있다.

즉, 각 1차 변수 x_j를 $m \geq 2$인 m-차원 단위 벡터, v_j로 대체하여 다음과 같은 문제를 정의하였다.

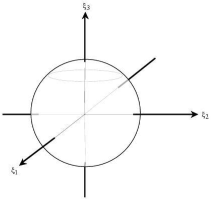


Figure: 각 x_j는 예를 들어 3차원 공간의 구면 S_3에 속하는 단위벡터 v_j로 대체 수 있다.

문제 18.5

$$\begin{array}{ll} \max & \frac{1}{2}\sum_{i<j} w_{ij}(1 - \mathbf{v}_i^T \mathbf{v}_j) \\ \text{s.t.} & \mathbf{v}_i \in S_m \ \forall i \in V \end{array}$$

문제 (18.3)의 임의의 가능해 $\bar{x}_j \ j \in V$가 주어졌을 때, $\bar{v}_j = (\bar{x}_j, 0, \cdots, 0) \ j \in V$은 (18.5)의 가능해가 되며, 두 해의 목적함수 값은 같음을 알 수 있다. 따라서 문제 18.5의 최적목적함수 값은 최대절단면문제의 상한이 됨을 알 수 있다. 이런 의미에서 문제 (18.5)는 문제 (18.3)의 완화문제(relaxation)가 된다.

단순히 완화문제를 만드는 것이 목적이라면 위에서 본 것처럼, $m \geq 2$인 어떤 m도 가능하다. 그러나 Goemans와 Williamson의 해법을 위해서는 $m = n$으로 놓아야한다.

문제 18.6

벡터계획법문제

$$\begin{aligned} \max \quad & \tfrac{1}{2}\textstyle\sum_{i<j} w_{ij}(1-\mathbf{v}_i^T\mathbf{v}_j) \\ \text{s.t.} \quad & \mathbf{v}_i \in S_n \ \forall i \in V \end{aligned} \tag{18.25}$$

우리는 여기서 자연스럽게 다음과 같은 두가지 질문을 하게 된다.

1. 벡터계획법문제 18.6는 효율적으로 풀 수 있는가?
2. 그렇다면, 문제 18.6의 최적해는 최대절단면문제의 '좋은' 해를 구하는데 사용될 수 있는가?

첫번째 질문의 답은 이미 오래전에 알려져 있었다. 다음과 같은 행렬들의 곱을 생각하자.

$$\begin{bmatrix} -- \mathbf{v}_1^T -- \\ -- \mathbf{v}_2^T -- \\ \vdots \\ -- \mathbf{v}_n^T -- \end{bmatrix} \begin{bmatrix} | & | & & | \\ \mathbf{v}_1 & \mathbf{v}_2 & \cdots & \mathbf{v}_n \\ | & | & & | \end{bmatrix} \tag{18.26}$$

$$= \begin{bmatrix} \mathbf{v}_1^T\mathbf{v}_1 & \mathbf{v}_1^T\mathbf{v}_2 & \mathbf{v}_1^T\mathbf{v}_3 & \cdots & \mathbf{v}_1^T\mathbf{v}_n \\ \mathbf{v}_2^T\mathbf{v}_1 & \mathbf{v}_2^T\mathbf{v}_2 & \mathbf{v}_2^T\mathbf{v}_3 & \cdots & \mathbf{v}_2^T\mathbf{v}_n \\ \vdots & & & & \vdots \\ \mathbf{v}_n^T\mathbf{v}_1 & \mathbf{v}_n^T \ v_2 & \mathbf{v}_n^T\mathbf{v}_3 & \cdots & \mathbf{v}_n^T\mathbf{v}_n \end{bmatrix}. \tag{18.27}$$

이 때, $y_{ij} \equiv \mathbf{v}_i^T\mathbf{v}_j (= \mathbf{v}_i \cdot \mathbf{v}_j)$로 정의하면
얻어진 (Gram) 행렬,

$$Y \equiv \begin{bmatrix} y_{11} & y_{12} & \cdots & y_{1n} \\ y_{21} & y_{22} & \cdots & y_{2n} \\ \vdots & & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nn} \end{bmatrix} \tag{18.28}$$

따라서 문제 18.6은 다음과 같은 SDP가 된다.

문제 18.7

$$\begin{array}{ll} \max & \frac{1}{2}\sum_{i,j} w_{ij}(1 - y_{ij}) \\ \text{s.t.} & y_{ii} = 1 \ \forall i = 1, \cdots, n \\ & Y = (y_{ij})\text{는 PSD.} \end{array}$$

PSD 행렬은 정방행렬의 곱으로 표시될 수 있으므로, $Y = W^T W$로 쓸 수 있고, W의 각 행으로 정의된 벡터들은 문제 18.6의 해가 된다. 따라서 문제 18.6는 SDP 문제 18.7과 동등한 문제가 되며, 이것은 문제 18.6가 다항시간에 풀 수 있다는 의미로서 첫번째 질문의 답이 된다.

이제 두번째 질문에 답해보자.

SDP완화를 이용한 근사해법
확률알고리듬

다음과 같은 확률알고리듬(randomized algorithm)을 생각해보자.

1. 문제 18.6의 최적해, $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n$를 구한다.
2. 단위 구면 S_n으로 부터 균일한 확률로 벡터, $\mathbf{r}$을 무작위로 선택한다.
3. $\mathbf{r}$과 수직이고 원점을 지나는 초평면으로 $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n$를 두 그룹으로 나눈후, 이에 따라 노드들을 양분하여 G의 절단면을 생성한다. 즉, 만약 $\mathbf{r}^T\mathbf{v}_i \geq 0$이면, $x_i \leftarrow 1$로 놓고, 그렇지 않으면, $x_i \leftarrow -1$로 놓는다.

정리 18.8

위의 해법은 최대절단면문제의 0.878-근사해법이 된다.

증명: 근사해법으로 얻어진 해의 기대값은 다음과 같이 주어진다.

$$E\left[w(S)\right] \tag{18.29}$$
$$= \sum_{i<j} w_{ij} \Pr\left[\mathbf{v}_i\text{와 } \mathbf{v}_j\text{가 } \mathbf{r}\text{과 수직인 초평면으로 나누어진다.}\right]$$
$$= \sum_{i<j} w_{ij} \Pr\left[\mathrm{sgn}(\mathbf{r}^T\mathbf{v}_i) \neq \mathrm{sgn}(\mathbf{r}^T\mathbf{v}_j)\right]$$

그러나, $\mathrm{sgn}(\mathbf{r}^T\mathbf{v}_i) \neq \mathrm{sgn}(\mathbf{r}^T\mathbf{v}_j)$일 필요 충분조건은, $\mathbf{r}$을 $\mathbf{v}_i$와 $\mathbf{v}_j$로 생성되는 평면에 투영했을 때, 한 벡터와는 예각을 다른 벡터와는 둔각을 이루는 것이다. (그림3 참고.)

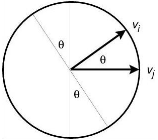


Figure: $\mathbf{v}_i$와 $\mathbf{v}_j$가 분리되는 경우.

이는 원주율 π 중에, $\mathbf{v}_i$와 $\mathbf{v}_j$의 사잇각의 두 배에 해당하는 부분에 $\mathbf{r}$이 투영될 확률과 같다. 따라서,

$$\begin{aligned} &\Pr[\mathbf{v}_i\text{와 } \mathbf{v}_j\text{가 } \mathbf{r}\text{과 수직인 초평면으로 나누어진다.}] \quad (18.30) \\ &= \frac{1}{\pi}\arccos(\mathbf{v}_i^T\mathbf{v}_j). \end{aligned}$$

그러므로,

$$\begin{aligned} E[w(S)] &= \textstyle\sum_{i<j} w_{ij}\frac{1}{\pi}\arccos(\mathbf{v}_i^T\mathbf{v}_j) && (18.31) \\ &\geq 0.878\tfrac{1}{2}\textstyle\sum_{i<j} w_{ij}(1-\mathbf{v}_i^T\mathbf{v}_j). && (18.32) \end{aligned}$$

(18.32)는 다음과 같은 사실때문에 성립힌다. (그림 4는 $\frac{\arccos x}{\pi}/(1/2)(1-x)$의 값을 $-1 \leq x \leq 1$의 범위에서 도시한 것이다.)

$$\frac{1}{\pi}\arccos x \geq 0.878\frac{1}{2}(1-x) \quad \forall \ -1 < x < 1 \qquad (18.33)$$

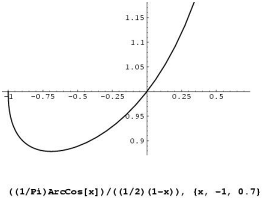


Figure: $\frac{\arccos x}{\pi}/(1/2)(1-x)$.

그러나, 벡터완화법문제 18.6의 최적목적함수값은, 최대절단면 문제의 목적함수값보다 크므로, 증명이 끝난다. □

0-1 2차함수계획법을 위한 SDP 완화

$0-1$ 2차 함수계획법문제은 가장 쉽게 SDP로 완화될 수 있음을 알 수 있는 문제이다. 1997년 Nesterov가 제안한, 근사값이 그 후에 개선된 해법들 보다는 나쁘지만 앞절의 방법을 간단하게 확장한 방법을 소개한다.

$0-1$ 2차 함수계획법문제는 간단한 변수치환을 거쳐, 다음과 같은 동등한 $-1, +1$ 2차 정수 계획문제로 쉽게 변환할 수 있다.

문제 18.9

$$\begin{array}{c} \max \sum_{i,j} a_{ij} x_i x_j \\ \text{s.t.} \quad x_i \in \{-1, 1\} \ \forall \ i \end{array}$$

여기서 $A = (a_{ij}) \in \mathcal{S}_n$는 PSD라고 가정할 수 있다. 그 이유는, $x_i^2 = 1$이므로 A의 대각선에 해당하는 부분은 상수 항에 해당하기 때문에 필요하다면 얼마든지 큰 상수를 대각선에 더할 수 있기 때문이다. 따라서 최적목적함수 값이 비음이라고 가정할 수 있다.

앞의 방법과 유사하게 x_i를 단위 벡터 $\mathbf{v}_i \in \mathbb{R}^n$로 치환하여, 다음과 같은 벡터계획법문제를 얻는다.

문제 18.10

$$\begin{array}{rl} \max \sum_{i,j} a_{ij}\mathbf{v}_i^T\mathbf{v}_j & \\ \text{s.t.} \quad \mathbf{v}_i^T\mathbf{v}_i = 1 \ \forall\ i & \\ \mathbf{v}_i \in \mathbb{R}^n & \end{array}$$

다음과 같은 확률알고리듬(randomized algorithm)을 생각해보자.

1. 위의 벡터계획법 문제의 최적해, $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_n$를 구한다.
2. 단위 구면 S_n으로 부터 균일한 확률로 벡터, $\mathbf{r}$을 무작위로 선택한다.
3. 최적해 $\mathbf{v}^*$로 부터 다음과 같이 $\bar{x}$를 구한다.

$$\bar{x}_i = \begin{cases} 1 & \text{if } \mathbf{r}^T\mathbf{v}_i^* \geq 0 \\ -1 & \text{if } \mathbf{r}^T\mathbf{v}_i^* < 0 \end{cases}$$

그러면 다음과 같은 근사값를 얻는다.

정리 18.11

위의 확률알고리듬은 $\frac{2}{\pi}$-근사해법이 된다.

$i \neq j$인 경우, 다음이 성립한다:

$$\Pr[\mathrm{sgn}(\mathbf{r}^T \mathbf{v}_i^* \neq \mathbf{r}^T \mathbf{v}_i^*)] = \Pr[\bar{x}_i \bar{x}_j = -1] = \frac{1}{\pi} \arccos \mathbf{v}_i^{*T} \mathbf{v}_j^*. \tag{18.34}$$

따라서, 다음과 같은 기대값을 얻는다:

$$\begin{aligned} & E[\bar{x}_i \bar{x}_j] \\ &= 1(1 - \tfrac{1}{\pi} \arccos \mathbf{v}_i^{*T} \mathbf{v}_j^*) + (-1)\tfrac{1}{\pi} \arccos \mathbf{v}_i^{*T} \mathbf{v}_j^* \\ &= 1 - \tfrac{2}{\pi} \arccos \mathbf{v}_i^{*T} \mathbf{v}_j^* \\ &= 1 - \tfrac{2}{\pi}(\tfrac{\pi}{2} - \arcsin \mathbf{v}_i^{*T} \mathbf{v}_j^*) \\ &= \tfrac{2}{\pi} \arcsin \mathbf{v}_i^{*T} \mathbf{v}_j^*. \end{aligned}$$

기본정리 18.12

만약 $z_{ij} = \arcsin x_{ij} - x_{ij}$ 이고 $|x_{ij}| \leq 1$ for all i, j 이면 $X \succeq 0$ 일 때, $Z \succeq 0$ 이 된다.

기본정리 18.12를 사용하면,

$$
\begin{aligned}
& E[\textstyle\sum_{i,j} a_{ij}\bar{x}_i\bar{x}_j] - \tfrac{2}{\pi}\sum_{i,j} a_{ij}\mathbf{v}_i^{*T}\mathbf{v}_j^* \\
&= \tfrac{2}{\pi}\textstyle\sum_{i,j} a_{ij}(\arcsin \mathbf{v}_i^{*T}\mathbf{v}_j^* - \mathbf{v}_i^{*T}\mathbf{v}_j^*) \\
&= \tfrac{2}{\pi}\textstyle\sum_{i,j} a_{ij}z_{ij} \\
&= \tfrac{2}{\pi} X \cdot Z \\
&\geq 0 \text{ (Lemma 18.12)}.
\end{aligned}
$$

따라서 정리가 증명된다. □

Part XIII

Chapter 28

근사적 Counting

정의 19.1

#P: L을 NP의 한 언어라고 하자. M을 이의 검증자(verifier)인 비결정튜링기계, 그리고 p를 예-근거의 길이의 다항함수 상한이라고 하자. 문자끈 $x \in \Sigma^*$에 대해, $f(x)$를 $|y| \leq p(|x|)$이며 $M(x, y)$가 받아들이는 y의 개수라고 하자. 이러한 함수 $f : \Sigma^* \longrightarrow \mathbb{Z}_+$의 집합을 #P라고 한다.

직관적으로 말하면 #P란 NP 문제의 문제예가 있을 때,'예'가 되는 "해"의 개수를 세는 문제라고 할 수 있다. 예를 들어, 무향그래프의 해밀톤 경로의 개수나, SAT의 충족 진리값 할당의 개수 등이 그 예이다. 따라서 #P는 최소한 NP 만큼 어려운 문제라고 할 수 있다.

직관적으로 말해서, 모든 #P 문제들이 어떤 #P 문제로 다항변환 가능할 때, 후자를 #P-complete이라고 한다. NP-complete 문제의 가능해를 세는 문제는 대부분 #P-complete이며, 흥미로운 것은 많은 P 문제의 가능해를 세는 문제 역시 #P-complete이라는 것이다.

정의 19.2

FPRAS: counting 문제가 $\#P$-complete 인 P에 속하는 문제가 있다고 하자. 이 때, 다음과 같은 알고리듬 $\mathcal{A}$를 fully polynomial time randomized approximation scheme (FPRAS)라고 한다: 모든 문자끈 $x \in \Sigma^*$와 오자 $\epsilon > 0$에 대해,

$$\Pr[|\mathcal{A}(x) - f(x)| \leq \epsilon f(x)] \geq \frac{3}{4}.$$

문제 19.3

DNF 해의 개수 $f = C_1 \vee C_2 \vee \cdots \vee C_m$을 n개의 부울변수 $x_1, \ldots, x_n$의 DNF 라고 하자. l_i를 리터럴이라고 할 때, 다음과 같이 표현되는 각 절 $C_i = l_1 \wedge l_2 \wedge \cdots \wedge l_{r_i}$은 가능해를 갖는다고 가정할 수 있다. (즉, 어떤 변수와 이의 부정이 같이 들어 있지 않다.) 우리의 문제는 f의 가능해의 개수 $\#f$를 계산하는 것이다.

기본적인 아이디어는 $\#f$에 대한 불편추정량 (unbiased estimator)이 되는 확률변수 X를 정의하는 것이다: $E[X] = \#f$. 만약에 그러한 X의 표준편차 $\sigma[X]$가 $E[X]$의 다항함수 배로 제한되면 FPRAS를 쉽게 만들수 있다: X의 값을 문제 입력 크기와 $\frac{1}{\epsilon}$의 다항함수 횟수 만큼 표본을 추출하여 평균을 출력하면 된다.

쉽게 생각할 수 있는 방법: 2^n개의 진리값 τ 위에 균일분포(uniform distribution)를 갖는 확률변수 Y를 다음과 같이 정의한다: τ가 f를 만족하면 $Y(\tau) = 2^n$, 그렇지 않으면 $Y(\tau) = 0$. 하지만 이 경우는 표준편차가 너무 커서 FPRAS를 이끌어 내지 못한다.

이러한 문제점을 극복하기 위해, f를 만족하는 진리값들만 0이 아닌 확률을 갖도록 확률변수를 정의한다.

S_i를 C_i를 만족하는 n개 변수의 진리값 집합이라고 하자. 그러면 $|S_i| = 2^{n-r_i}$, $\#f = |\bigcup_i^m S_i|$. $c(\tau)$를 τ가 만족하는 절의 개수라고 하자. M을 S_i들이 '중복을 모두 포함하는' 합집합이라고 하자: $|M| = \sum_i^m |S_i|$. M은 각 τ를 $c(\tau)$ 횟수 포함하게 된다.

f를 만족하는 진리값 τ를 $c(\tau)/|M|$의 확률로 선택되로록 한다. 그리고 $X(\tau) = |M|/c(\tau)$로 정의하자.

기본정리 19.4

확률변수 X를 효율적으로 표본수집이 가능하며 $\#f$에 대한 불편추정량이 된다.

증명: 우선 각 τ가 $c(\tau)/|M|$의 확률로 선택되도록 효율적인 표본수집이 가능하다는 것을 보이자. 우선 각 절, C_i가 $|S_i|/|M|$의 확률로 절을 선택한다. 그리고 이 절을 만족하는 진리값 집합 중에 균등한 확률로 하나를 선택한다. 그러면, 우리가 원하는 확률의 표본수집이 된다는 것을 쉽게 확인할 수 있다 (해볼것).

또한,

$$
\begin{aligned}
E[X] &= \textstyle\sum_\tau \Pr[r \text{ 이 선택됨}] \cdot X(\tau) \\
&= \textstyle\sum_{f \text{ 를 만족하는 } \tau} \frac{c(\tau)}{|M|} \frac{|M|}{c(\tau)} = \#f. \ \square
\end{aligned}
$$

기본정리 19.5

m을 절의 개수라고 하면, $\frac{\sigma(X)}{E[X]} \leq m - 1$.

증명: $\frac{|M|}{m}$을 α라고 하자. 그러면 $E[X] \geq \alpha$가 성립한다. 각 진리값에 대해, $1 \leq c(\tau) \leq m$. 따라서 $X(\tau) \in [\alpha, m\alpha]$, 따라서 $|X(\tau) - E[X]| \leq (m-1)\alpha$. 따라서, 표준편차, $\sigma(X) \leq (m-1)\alpha$.
□

기본정리 19.6

k개의 표본의 평균을 Y_k라고 임의의 $\epsilon > 0$에 대하여, $k = 4(m-1)^2/\epsilon^2$이면

$$\Pr[|Y_k - \#f| \leq \epsilon \#f] \geq \frac{3}{4}.$$

증명: 체비셰프의 부등식, $\sigma(Y_k) = \sigma(X)/\sqrt{k}$를 사용하면,

$$\Pr[|Y_k - E[Y_k]| \geq \epsilon E[Y_k]]$$
$$\leq \left(\frac{\sigma(Y_k)}{\epsilon E[X_k]}\right)^2 = \left(\frac{\sigma(X)}{\epsilon \sqrt{k} E[X]}\right)^2 \leq \frac{1}{4} \quad \square$$

이로서 FPRAS가 완성된다.

Part XIV

Chapter 29

NP-complete임을 증명하기

From *Computers and intractability* by Garey and Johnson

결정문제(decision problem) Π가 NP-complete임을 증명하는 단계

(1) $\Pi \in \mathrm{NP}$임을 보인다: 답이 “Yes” 일 때, 이를 다항시간에 확인 할 수 있는 방법, 즉 yes-certificate (증명, proof, 또는 verifier)이 존재함을 보인다.

(2) Π를 이미 NP-complete임을 알고 있는 문제 Π'으로 다항시간에 변환이 가능한 것을 보인다.

Cook의 정리에 의하여 SAT를 Π'으로 선택할 수 있다. 이를 시작으로 다음의 6개 문제의 NP-completeness를 단계적으로 증명한다.

문제 20.1

SAT
입력: 유한개의 부울변수 집합 $X = \{x_1, x_2, \ldots, x_n\}$의 리터럴들의 disjunction으로 구성된 절들의 집합 $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$.
질문: 모든 절을 만족하는 진리값이 존재하는가?

문제 20.2

3SAT
입력: 유한개의 부울변수 집합 $X = \{x_1, x_2, \ldots, x_n\}$의 세 개의 리터럴들의 disjunction으로 구성된 절들의 집합 $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$.
질문: 모든 절을 만족하는 진리값이 존재하는가?

문제 20.3

3DM

입력: 각 q개의 원소를 갖는 서로 소 집합 W, X, 그리고 Y. 부분집합 $M \subseteq W \times X \times Y$.

질문: M은 완전짝짓기 M'을 갖는가? 즉, 같은 좌표에 같은 원소를 갖지 않은 q개의 순서쌍 집합 M'이 M에 존재하는가?

문제 20.4

VC

입력: 무향그래프 $G = (V, E)$와 양수 $k(\leq |V|)$.

질문: G에 크기가 k 이하인 vertex cover C가 존재하는가? 즉, E의 모든 호의 끝 마디 중 적어도 한 개를 포함하며, 크기가 k를 넘지 않는 V의 부분집합 C가 존재하는가?

문제 20.5

CLIQUE
입력: 무향그래프 $G = (V, E)$와 양수 $k(\leq |V|)$.
질문: G에 크기가 k 이하인 클릭 Q 존재하는가? 즉, Q로 유도된 G의 부분그래프 $G[Q]$가 완전호그래프가 되며, 크기가 k와 같거나 큰 V의 부분집합 Q가 존재하는가?

문제 20.6

HC (Hamiltonian circuit)
입력: 무향그래프 $G = (V, E)$.
질문: G가 해밀턴 회로(Hamiltonian circuit)를 가지는가? 즉, 모든 마디를 포함하는 단순회로를 부분그래프로 가지는가 ?

문제 20.7

PARTITION
입력: 유한집합 A와 그 원소 $a \in A$의 가중치 $w : A \to \mathbb{Z}_+$.
질문: 가중치의 합이 같도록 A를 두개의 집합으로 나눌수 있는가 ?

정리 20.8

3SAT은 NP-*complete이다.*

증명: 3SAT $\in$ NP임은 쉽게 알 수 있다: 모든 절을 만족하는 진리값을 yes-certificate으로 사용하면 O(절의 수)의 시간에 확인이 가능하다. (단, 그러한 진리값을 어떻게 구하는지는 *요점이 아니라는* 것을 명심하자.)
SAT $\propto$ 3SAT: 주어진 SAT의 한 절 c가 k개의 리터럴 $z_1, \cdots, z_k$로 이루어졌다고 하자: $k \geq 4$인 경우 $c := z_1 \vee z_2 \vee \cdots \vee z_k$. c를 리터럴을 세개씩 가진 절로 변환하면 된다.
$c' = (z_1 \vee z_2 \vee y_1) \wedge (\bar{y}_1 \vee z_3 \vee y_2) \wedge (\bar{y}_2 \vee z_4 \vee y_3) \wedge \cdots (\bar{y}_{k-4} \vee z_{k-1} \vee y_{k-3}) \wedge (\bar{y}_{k-3} \vee z_{k-1} \vee \vee z_k)$.
c와 c'의 satisfiability가 동치라는 것을 아래를 증명하는 것으로 시작하여 귀납적으로 확인 할 것.

$$z_1 \vee \cdots \vee z_m \equiv (z_1 \vee \cdots z_l \vee y) \vee (z_{l+1} \vee \cdots \vee z_m \vee \bar{y}).$$

□

연습문제 20.9

$k \le 2$인 경우, c'을 어떻게 구성할 수 있는가?

연습문제 20.10

2SAT은 다항시간에 풀 수 있음을 보여라.

정리 20.11

3DM은 NP-*complete이다.*

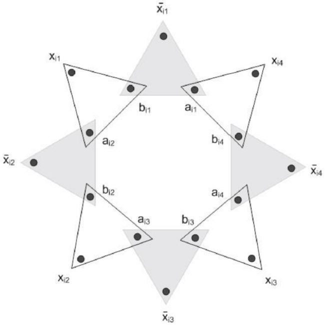
증명: 3SAT $\propto$ 3DM: 부울변수 집합 $X = \{x_1, x_2, \ldots, x_n\}$과 그 리터럴들로 구성된 절들의 집합 $C = \{c_1, c_2, \ldots, c_m\}$을 갖는 3SAT을 생각 하자. 정확히, 3SAT을 만족하는 진리값이 존재할 때, 완전짝짓기를 갖는 집합 W, X, Y와 부분집합 $M \subseteq W \times X \times Y$를 구성하면 된다.
좀더 구체적으로 말해, 3SAT (또는 SAT)의 모든 절을 만족하는 진리값이 존재한다는 것은 **같은 부울변수의 원형과 부정형이 동시에 나타나지 않도록 각 절에서 리터럴을 한개씩 선정할 수 있다는 것**과 동치라는 것을 명심하자. 이 것은 해당 변수가, 사용된다면, 모든 절에서 True 혹은 False 한 가지 경우로만 사용되어야 한다는 것과 동치이다.

이러한 조건의 성립과 일치하는 3DM 문제 순서쌍들의 집합을 세 단계로 구성할 것이다.

단계 I에서는 어떤 부울변수도 원형과 부정형이 동시에 두 개 이상의 절에서 선택이 되지 않도록 순서쌍을 구성한다. 단계 II에서는 단계 I의 규칙에 맞는 각 절 한 개 리터럴 선택이 가능하다는 것과 가능한 순서쌍이 존재한다는 것이 동치가 되도록 순서쌍 집합을 구성할 것이다. 마지막으로 단계 III에서는, 단계 I과 II의 순서쌍이 존재하는 경우, 반드시 완전 짝짓기로 확장되도록 순서쌍을 첨가 할 것이다.

I. 우선 각 변수 x_i에 대하여 다음과 같은순서쌍들을 모든 절 $j = 1, 2, \ldots, m$에 대해 구성한다:

$$T_i^t = \{(\bar{x}_{ij}, a_{ij}, b_{ij}) : 1 \leq j \leq m\}$$
$$T_i^f = \{(x_{ij}, a_{i(j+1)}, b_{ij}) : 1 \leq j < m\} \cup \{(x_{im}, a_{i1}, b_{im})\}$$



각 변수 x_i에 ($i=1,\cdots,n$) 대해 하나씩 구성, $m=4$

a와 b는 위를 제외한 순서쌍에는 나타나지 않는 '내부용' 변수이다. 따라서, 완전짝짓기를 위해서는, 각 i에 대하여 위에서 정확히 m개가 선택되어야 하며, 이는 모두 T_i^t의 원소들이거나 T_i^f의 원소들이라는 것을 알 수 있다.

이는 x_i 값을 모든 절에서 True와 False로 놓는 것에 대응하게 된다. 이와 동치로, 모든 절이 자신을 만족시키는 리터럴을 한 개씩 선택할 때, 어떤 부울변수 x_i도 원형과 부정형이 동시에 두 개 이상의 절에서 선택되지 않는 것을 의미한다.

II. 각 절 c_j에 두 내부용 변수를 대응시킨다: $s_{1j} \in X$, $s_{2j} \in Y$. 그리고 다음의 순서쌍을 구성한다: $C_j = C_{j1} \cup C_{j2}$,
$C_{j1} = \{(x_{ij}, s_{1j}, s_{2j}) : x_i \in c_j\}$, $C_{j1} = \cup\{(\bar{x}_{ij}, s_{1j}, s_{2j}) : \bar{x}_i \in c_j\}$.

따라서, 완전짝짓기가 되기 위해서는 C_j에서는 오직 한 개의 순서쌍을 선택하게 된다. 이는 각 절이 자신을 만족시키는 "대표" 리터럴을 하나 선택하는 것에 대응된다. 만약 절 j가 x_{ij}를 선택하였다면 이는 원형 리터럴 x_i를 선택하였다는 의미이며, 단계 I의 순서쌍 중에서 T_i^t가 선택되는 경우에만 가능하다. 따라서 I의 규칙에 따라, 다른 절들도, x_i를 선택한다면 역시 같은 원형으로만 선택할 수 있다. 마찬가지로 만약 C_j에서 $\bar{x}_{ij}$가 선택된다면 이는 부정형 $\bar{x}_i$를 자신을 만족시키는 "대표" 리터럴로 선택한다는 의미이며, 단계 I에서 T_i^f가 선택된 경우에만 가능하다. 그리고 x_i는 다른 절에서도 사용된다면, 오직 부정형으로만 사용된다.

III. 주어진 3SAT을 만족하는 진리값이 존재하는 것과, I에서 구성한 집합에서 정확히 mn개 그리고 II에서 구성한 집합에서는 m개의 순서쌍을, 같은 위치에 같은 원소가 발생하지 않도록, 선택할 수 있는 것과 동치라는 것을 알 수 있다.
이 순서쌍의 첫째 원소의 집합 W를 x_{ij} 또는 $\bar{x}_{ij}$의 집합으로 놓으면 모두 $2mn$개이다. 두 번째 원소의 집합 X가 a_{ij}와 s_{1j}들을 포함하게 하고, 세 번째 원소의 집합 Y가 b_{ij}와 s_{2j}들을 포함하게 하면, 앞에서 언급한 3SAT을 만족하는 진리값에 해당하는 순서쌍들의 집합은 $mn + m$개가 된다.
따라서 이러한 순서쌍 집합을 완전 짝짓기로 항상 확장할 수 있게 하기 위해서 나머지 $2mn - mn - m$개의 x_{ij} 또는 $\bar{x}_{ij}$가 포함되는 순서쌍이 항상 존재하도록 다음과 같은 순서쌍들을 M에 포함시키면 된다. 이는 각 x_{ij} 또는 $\bar{x}_{ij}$에게 $2mn - mn - m$개의 2원소 순서쌍 (g_{1k}, g_{1k}), $1 \le k \le m(n-1)$가 모두 연결 가능하도록 다음과 같은 순서쌍 집합을 M에 포함시키는 것이다.
$G = \{(x_{ij}, g_{1k}, g_{2k}), (\bar{x}_{ij}, g_{1k}, g_{2k}) : 1 \le k \le m(n-1),\ 1 \le i \le m,\ 1 \le j \le n.\}$
이는 X에는 g_{1k}, $1 \le k \le m(n-1)$가 Y에는 g_{2k}, $1 \le k \le m(n-1)$가 추가되는 것을 의미한다. □

연습문제 20.12

3DM $\propto$ 3SAT.

문제 20.13

Exact cover by 3-sets (X3C)
입력: 크기가 $3q$인 집합 X와 X의 3-원소 집합의 집합 C.
질문: X의 각 원소가 C'의 꼭 한개의 집합에만 나타나는 부분집합 C'이 C에 존재하는가?

연습문제 20.14

3DM $\propto$ 3XC.

정리 20.15

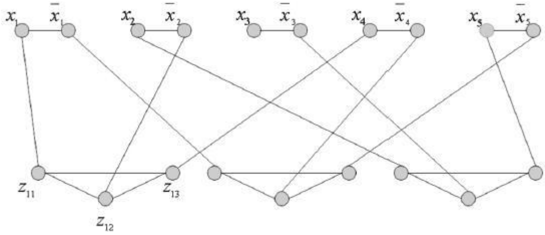
3SAT $\propto$ VC

증명: 3SAT $\propto$ 3DM: 부울변수 집합 $X = \{x_1, x_2, \ldots, x_n\}$과 그 리터럴들로 구성된 절들의 집합 $C = \{c_1, c_2, \ldots, c_m\}$을 갖는 3SAT을 생각 하자. 각 절의 세 개의 리터럴들을 임의의 순서로 고정하고, j 번째 절의 l 번째 리터럴을 z_{jl}로 표기하자:

$c_1 = z_{11} \vee z_{12} \vee z_{13}, \ldots, c_m = z_{m1} \vee z_{m2} \vee z_{m3}$.

$k = n + 2m$.

$(x_1 \vee \bar{x}_2 \vee x_4) \wedge (\bar{x}_1 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (x_2 \vee \bar{x}_3 \vee x_5)$



□

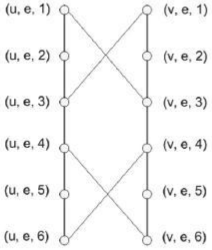
연습문제 20.16

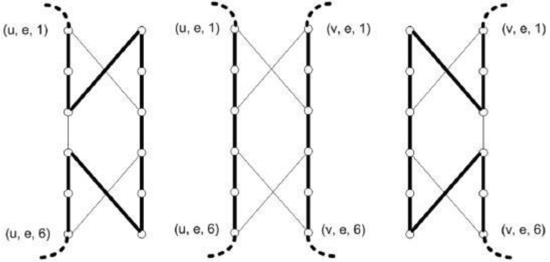
VC $\propto$ STABLE $\propto$ CLIQUE $\propto$ VC

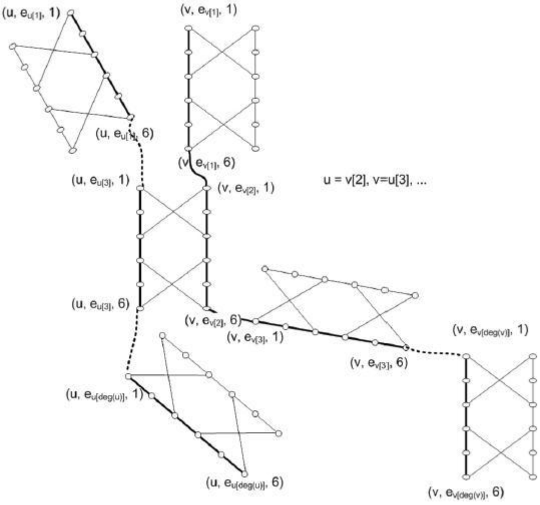
정리 20.17

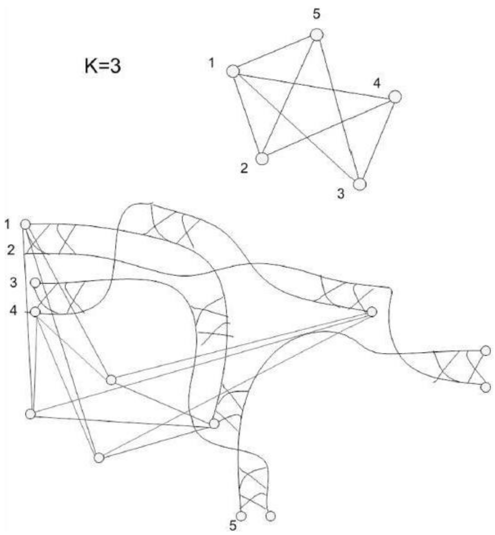
VC $\propto$ HC.

증명: 임의의 VC 문제예를 생각하자: 무향그래프 $G = (V, E)$와 자연수 $k \leq |V|$. 이에 따라, 정확히 G가 k를 넘지 않는 vertex cover를 가질 때, 해밀턴 회로를 갖는 그래프 $G' = (V', E')$를 다항시간에 구성해야 한다.


(u, e, 1)
(v, e, 1)
(u, e, 2)
(v, e, 2)
(u, e, 3)
(v, e, 3)
(u, e, 4)
(v, e, 4)
(u, e, 5)
(v, e, 5)
(u, e, 6)
(v, e, 6)


(u, e, 1)
(u, e, 6)
(u, e, 1)
(v, e, 1)
(u, e, 6)
(v, e, 6)
(v, e, 1)
(v, e, 6)

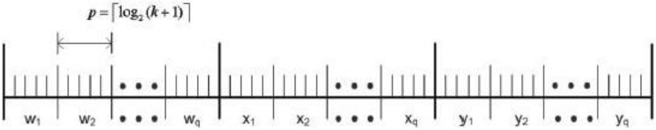

(u, e_u[1], 1)
(u, e_u[1], 6)
(v, e_v[1], 1)
(v, e_v[1], 6)
(u, e_u[3], 1)
(u, e_u[3], 6)
(v, e_v[2], 1)
(v, e_v[2], 6)
u = v[2], v=u[3], ...
(v, e_v[3], 1)
(v, e_v[3], 6)
(u, e_u[deg(u)], 1)
(u, e_u[deg(u)], 6)
(v, e_v[deg(v)], 1)
(v, e_v[deg(v)], 6)


K=3
5
1
4
2
3
1
2
3
4
5

정리 20.18

3DM $\propto$ PARTITION.

증명: 임의의 3DM 문제 예, 즉, 각 q개의 원소를 갖는 서로 소 집합 $W = \{w_1, w_2, \ldots, w_q\}$, $X = \{x_1, x_2, \ldots, x_q\}$, 그리고 $Y = \{y_1, y_2, \ldots, y_q\}$와 부분집합 $T \subseteq W \times X \times Y$를 생각하자. $|T| = k$, $T = \{t_1, t_2, \ldots, t_k\}$라고 하자.
정확히 M이 완전짝짓기를 가지는 경우 가중치의 합이 같은 분할을 갖는 PARTITION의 문제 예를 만들어 보자:
$S = \{s(a) : a \in A\}$.
S는 $k+2$ 개의 원소를 갖는데, 이 중 k개, $\{a_1, a_2, \ldots, a_k\}$는 T의 각 원소에 대응되며 다음 그림과 같이 $3q \times p$개의 자릿수를 갖는 2진법 가중치 $s(a_i)$를 갖는다.



만약 $t_i = (w_{j_i}, x_{k_i}, y_{l_i})$이라면 이에 대응하는 a_i는 그림의 w_{j_i} 칸, x_{k_i} 칸, 그리고 z_{l_i} 칸들의 가장 오른 쪽, 즉 가장 작은 자릿수들을 모두 1로 하고 나머지는 모두 0으로 놓은 수를 가중치 $s(a_i)$로 정의한다:

$$s(a_i) = 2^{p(3q-j_i)} + 2^{p(2q-k_i)} + 2^{p(q-l_i)}.$$

따라서 $T' = \{t_{i_1}, t_{i_2}, \ldots, t_{i_q}\} \subseteq T$가 완전짝짓기가 될 필요충분조건은 T'의 어떤 두 가중치도 같은 칸 가장 오른쪽 자릿수에 똑같이 1을 갖지 않는다는 것으며, 또한 모든 칸이 가장 오른쪽 자릿수에 1을 갖는다는 것이다.
각 칸의 자릿수의 수가 $p = \lceil \log_2(k+1) \rceil$이기 때문에 k개의 가중치를 모두 더해도 어떤 칸의 숫자들의 합이 왼쪽 칸으로 올라가는 일은 발생하지 않는다.
따라서, 완전짝짓기가 될 필요충분조건은 그 대응하는 가중치의 합이 다음과 같다는 것이다:

$$\sum_{j=1}^{q} s(a_{i_j}) = B.$$

나머지 두 개의 원소 b_1과 b_2는 다음과 같이 정의하자:

$$s(b_1) = 2\sum_{i=1}^{k} s(a_i) - B, \; s(b_2) = 2\sum_{i=1}^{k} s(a_i) + B.$$

그러면, $\cdots$.

□

몇가지 테크닉

- Restriction
- Local Replacement
- Component Design

Restriction

SET COVER
HITTING SET
SUBGRAPH ISOMORPHISM
BOUNDED DEGREE SPANNING TREE
MINIMUM EQUIVALENT DIGRAPH
KNAPSACK
MULTIPROCESSOR SCHEDULING

Local Replacement

문제 20.19

ENSEMBLE COMPUTATION
입력: 유한집합 A, 그 부분집합들의 집합 $\mathcal{C}$, 그리고 자연수 J.
질문: 다음의 조건을 만족하는 아래와 같은 열(sequence)이 존재하는가?

$$z_1 = x_1 \cup y_1,\ z_2 = x_2 \cup y_2, \ldots,\ z_j = x_j \cup y_j.$$

1. $j \leq J$,
2. x_i와 y_i는 각각 A의 원소 a로 만들어진 집합 $\{a\}$이거나 $k < i$인 z_k이고, x_i와 y_i는 서로 소이며,
3. 모든 $c \in \mathcal{C}$가 z_i 중에 나타난다.

정리 20.20

VC $\propto$ ENSEMBLE COMPUTATION.

증명: 그래프 $G = (V, E)$와 자연수 K로 정의된 VC를 EC로 변환하자. By a "local replacement":

$$A \leftarrow V \cup \{a_0\}, \; C \leftarrow \{\{a_0, u, v\} : uv \in E\}, \; J \leftarrow K + |E|.$$

$\square$

문제 20.21

PARTITION INTO TRIANGLES

입력: 어떤 자연수 q에 대해 $|V| = 3q$인 그래프 $G = (V, E)$.

질문: V를 서로 소인 3원소 집합(3-sets) $V_1, V_2, \ldots, V_q$로 분할하여 V_i로 유도된 그래프가 모두 삼각형(C_3)이 되도록 할 수 있는가?

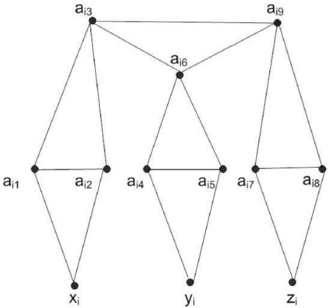
정리 20.22

X3C $\propto$ PIT.

증명: 어떤 자연수 q에 대해 $|X| = 3q$인 집합 X와 그 부분집합들의 집합 $\mathcal{C}$로 정의된 X3C를 생각하자.

$\mathcal{C}$의 집합 $c_i = \{x_i, y_i, z_i\}$를 다음과 같은 그래프 E_i로 대체하여 전체 그래프를 다음과 같이 정의,

$$V := X \cup \bigcup_{i=1}^{|\mathcal{C}|} \{a_{ij} : j = 1, \ldots, 9\}, \; E = \bigcup_{i=1}^{|\mathcal{C}|} E_i.$$



□

L-변환과 Max-SNP-hardness

First-order logic의 Syntax

정의 20.23

기호집합(vocabulary) $\Sigma = (\Phi, \Pi, r)$은 *함수 기호* (function symbols) 집합 Φ, *관계 기호*(relation symbols) 집합 Π, 그리고 각 함수 기호와 관계 기호에게 비음 정수를 대응시키는 '*arity*' 함수 r로 이루어진다.
'arity' 함수 r은 각 함수 및 관계 기호가 몇 개의 변수를 사용하는가를 나타낸다. 함수 $f \in \Phi$가 $r(f) = k$이면 k-ary 함수 기호라고 하고, 관계 기호 $R \in \Pi$가 $r(R) = k$이면 역시 k-ary라고 한다. 0-ary 함수는 상수(constant)라고 한다. 관계기호는 0-ary가 될 수 없다. 그리고 관계기호 집합 Π 중에 반드시 등호 $=$ 가 포함되어 있다. 그리고 고정된 가산 집합인 변수 집합 $\{x, y, z, \ldots\}$이 주어지며, 이들은 경우에 따라 정의된 전체집합(universe)에서 값을 취한다.

정의 20.24

항(terms) 기호집합(vocabulary) Σ를 사용하여 다음과 같이 항(term)들을 정의한다. 우선 각 V의 각 변수는 모두 항이다. 만약 $f \in \Phi$가 k-ary 함수 기호이고, $t_1, t_2, \ldots, t_k$가 각각 항이면, $f(t_1, \ldots, t_k)$는 항이다. ($k = 0$로 잡으면, 상수 $c \in \Phi$ 역시 항이 되는 것을 알 수 있다.)
항이 정의되면, Σ 상의 논리식(expression)을 정의할 수 있다. 만약 $R \in \Pi$가 k-ary 관계 기호이고 $t_1, t_2, \ldots, t_k$가 각각 항이면, $R(t_1, \ldots, t_k)$를 기본논리식 (atomic expression) 이라고 한다. 일차 논리식(first-order expression)은 다음과 같이 정의한다:

1. 기본논리식은 모두 일차논리식이다.
2. 만약 ϕ와 ψ가 일차논리식이면 $\neg\phi$, $(\phi \vee \psi)$, 그리고 $(\phi \wedge \psi)$ 역시 일차논리식이 된다.
3. 만약 ϕ가 일차 논리식이고 x가 임의의 변수이면 $(\forall x \phi)$는 일차논리식이다.

다음과 같은 표기법을 정의한다.

$$\exists x \phi \equiv \neg(\forall x \neg \phi)$$

예 20.25

수론 $\Sigma_{\mathbb{N}} = (\Phi_{\mathbb{N}}, \Pi_{\mathbb{N}}, r_{\mathbb{N}})$,
$\Phi_{\mathbb{N}} = \{0, \sigma, +, \times, \uparrow\}$.
0은 상수, σ는 unary 함수로 다음 수를 대응 시킨다; 합 +;, 곱 $\times$; 지수승 $\uparrow$
$\Pi_{\mathbb{N}} = \{=, <\}$.
$\forall x < (+(x, \sigma(\sigma(0))), \sigma(\uparrow (x, \sigma(\sigma(0))))$
또는
$\forall x (x + 2) < \sigma((x \uparrow 2))$

Max-SNP-hardness

정의 20.26

다음과 같이 표현되는 구조 G에 대한 모든 술어논리식(predicates)의 집합을 NP라고 한다:

$$\exists S \phi(G, S).$$

여기서, S는 하나의 구조(structure)이고 ϕ가 일차(first order)이다.

정의 20.27

다음이 성립하면 최대화문제 Π를 Max-SNP에 속한다고 한다.

$$\Pi = \max_S |\{x : \Psi(x, G, S) = \mathrm{True}\}|$$

여기서, Ψ는 수량사 $\exists$ 또는 $\forall$를 갖지 않는 술어논리식 (predicates)이다.

예 20.28

MAX 3SAT $\in$ Max-SNP:

$$\text{MAX 3SAT} = \max_S |\{(x_1, x_2, x_3) :$$
$$\{(x_1, x_2, x_3) \in C_0 \rightarrow (x_1 \in S \vee x_2 \in S \vee x_3 \in S)\} \wedge$$
$$\cdots \wedge \{(x_1, x_2, x_3) \in C_3 \rightarrow (x_1 \notin S \vee x_2 \notin S \vee x_3 \notin S)\}\}|$$

(x_1, x_2, x_3)는 (변수가 아니라) 각 절의 리터럴을 상징하는 벡터, S는 변수 값이 참인 변수들의 집합, C_i는 i개의 부정형 리터럴을 갖는 모든 절의 집합이다.

예 20.29

MAX CUT $\in$ Max-SNP:

$$\text{MAX CUT} = \max_S |\{(u, v) :$$
$$(u < v) \wedge E(u, v) \wedge S(u) \neq S(v)\}|$$

S는 절단면에 의해 나뉘는 두 마디 집합을 표현하는 0-1 벡터로 u가 S에 속하면 $S(u) = 1$, 그렇지 않으면 $S(u) = 0$. 호 uv가 존재하면 $E(u, v) = 1$, 그렇지 않으면 $E(u, v) = 0$.

정리 20.30

Max-SNP의 최대화문제는 어떤 상수 ϵ에 대해 $(1-\epsilon)$-근사해법을 갖는다.

정의 20.31

다음과 같은 다항시간 알고리듬 f와 g, 상수 $\alpha, \beta > 0$이 존재할 때, 최적화(최대, 또는 최소)문제 Π_1이 Π_2로 *L*-변환된다고 한다: Π_1의 임의의 예 I_1에 대하여,

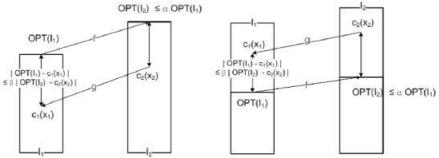
1. f는, $\mathrm{OPT}(I_2) \leq \alpha \mathrm{OPT}(I_1)$를 만족하는 Π_2의 예 I_2를 생성한다.
2. g는, I_2의 임의의 해 x_2에 대하여, $|c_1(x_1) - \mathrm{OPT}(I_1)| \leq \beta |c_2(x_2) - \mathrm{OPT}(I_2)|$인 I_1의 해 x_1을 생성한다.

성질 20.32

L-변환의 합성은 다시 L-변환이 된다.

성질 20.33

Π_1이 Π_2로 L-변환되고, Π_2의 오차가 $\epsilon > 0$인 근사해법이 존재하면 Π_1은 오차가 $\alpha\beta\epsilon$인 근사해법이 가능하다. (여기서, Π_1과 Π_2는 각각 최소 또는 최대화 문제가 될 수 있다.)



정의 20.34

Max-SNP의 모든 문제가 최적화 문제 Π로 L-변환될 때, Π를 Max-SNP-hard라고 한다. 이 때, Π 역시 Max-SNP에 속하면 Max-SNP-complete이라고 한다.

정리 20.35

MAX 3SAT은 Max-SNP-complete이다.

간격보존변환과 근사불능성

SAT 계열의 문제는 근사불능성의 증명에도 중요한 역할을 한다. 다음과 같은 3SAT의 최적화 문제를 생각하자.

정의 20.36

MAX3SAT
문제예 I: n개의 부울변수 $x = (x_1, x_2, \ldots, x_n)$의 세개 리터럴들의 disjunction으로 이루어진 절(clause) c들의 집합 $\mathcal{C}$의 cnf,

$$f = \bigwedge_{c \in \mathcal{C}} c.$$

최적해: f의 만족되는 절들의 비율 MAX3SAT(I)가 최대가 되는 x의 진리값.

정리 20.37

다음을 만족하는 상수 $\epsilon > 0$과 SAT으로부터 MAX3SAT으로의 다항변환 τ가 존재한다: 임의의 SAT의 문제 예 I에 대하여,

$$I \in \mathrm{SAT} \Rightarrow \mathrm{MAX3SAT}(\tau(I)) = 1,$$
$$I \notin \mathrm{SAT} \Rightarrow \mathrm{MAX3SAT}(\tau(I)) < \frac{1}{1+\epsilon}.$$

증명: 나중에. □

따름정리 20.38

MAX3SAT의 $(1+\epsilon)$-근사 알고리듬은 ($\mathrm{P} \neq \mathrm{NP}$ 이면) 불가능하다.

이러한 결과는 다음과 같은 간격보존변환(gap-preserving reduction)을 사용하여 다양한 최적화문제의 근사불능성을 증명할 수 있다.

정의 20.39

간격보존변환: Π와 Π'을 최대화문제라고 하자. 다음을 만족하는 Π로 부터 Π'으로의 다항변환 f를 파라미터 (c, ρ), (c', ρ')을 갖는 간격보존변환이라고 한다: 임의의 $I \in \Pi$에 대하여,

$$\mathrm{OPT}(I) \geq c \Rightarrow \mathrm{OPT}(f(I)) \geq c',$$
$$\mathrm{OPT}(I) < \frac{c}{\rho} \Rightarrow \mathrm{OPT}(f(I)) < \frac{c'}{\rho'}.$$

여기서 c와 ρ는 $|I|$의 c'와 ρ'은 $|f(I)|$의 함수이며 $\rho, \rho' \geq 1$이다.

만약 위와 같은 간격보존변환 f를 갖는 Π와 Π'이 있고, 다음과 같은 SAT로 부터 Π로 다항변환 τ가 증명되었다고 하자:

$$I \in \mathrm{SAT} \Rightarrow \mathrm{OPT}(\tau(I)) \geq c,$$
$$I \notin \mathrm{SAT} \Rightarrow \mathrm{OPT}(\tau(I)) < \frac{c}{\rho}.$$

그러면,

$$I \in \mathrm{SAT} \Rightarrow \mathrm{OPT}(f(\tau(I))) \geq c',$$
$$I \notin \mathrm{SAT} \Rightarrow \mathrm{OPT}(f(\tau(I))) < \frac{c'}{\rho'},$$

즉, Π'의 ρ'-근사 해법은 불가능하다는 것을 의미한다.