

アルゴリズム論 Theory of Algorithms

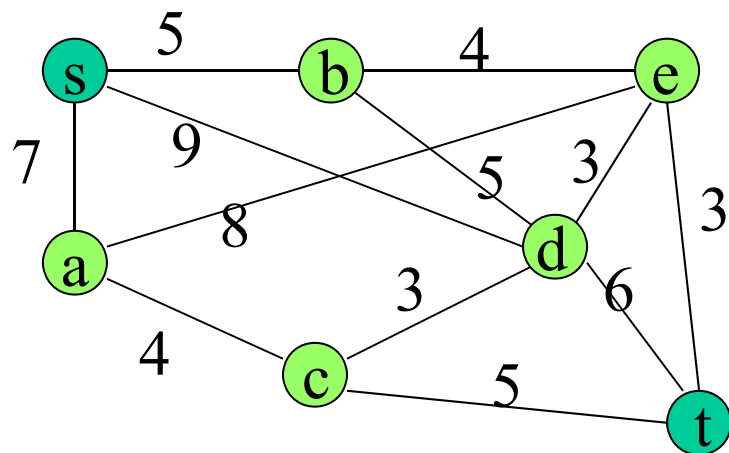
第4回講義 貪欲アルゴリズム ～続き～

アルゴリズム論 Theory of Algorithms

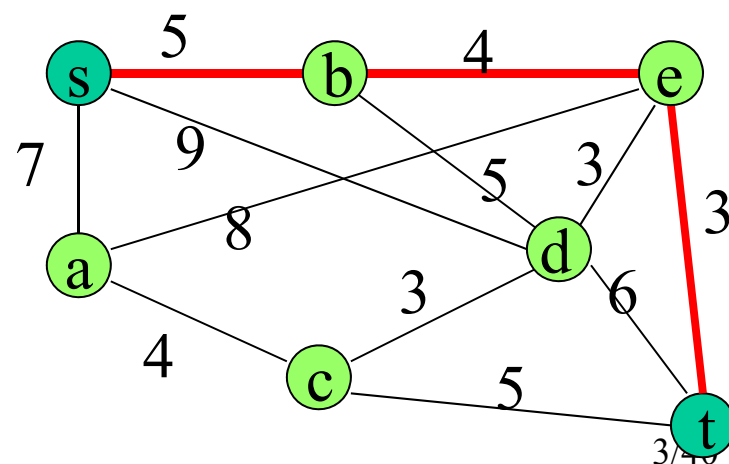
Lecture #4 Greedy Algorithm ～Continued～

問題P14:(最短経路問題)

n 都市を結ぶ道路網が重みつきグラフの形で与えられているとき、任意に2都市を結ぶ最短経路(重み最小の経路)を求めよ.

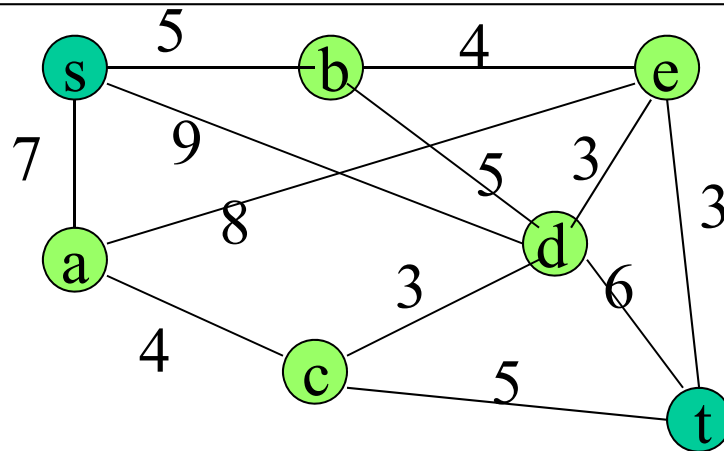


sからtへの最短経路？

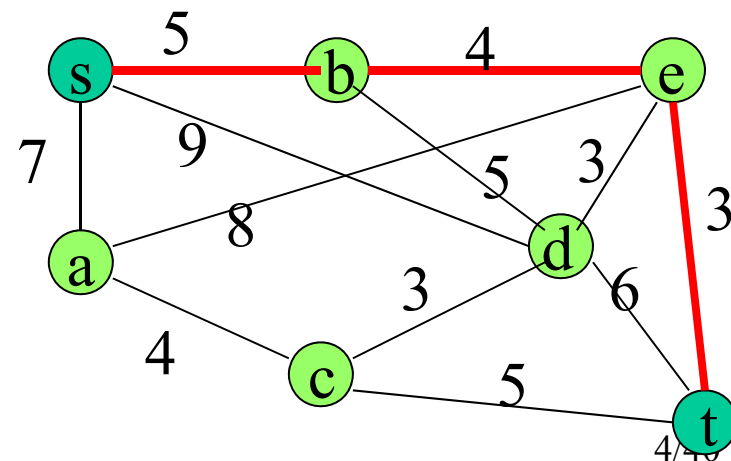


Problem P14: (Shortest Path Problem)

Given a road network connecting n cities as a weighted graph, find a shortest path (of minimum weight) between arbitrarily specified two cities.



Shortest path from s to t?



ダイクストラのアルゴリズム

始点 s から終点 t までの最短経路だけでなく, s から他の全ての頂点への最短経路を求める.

$D[v]$: s から頂点 v までの最短経路の長さ
として, すべての頂点について $D[]$ の値を求める.

貪欲選択:

最初は

$$D[s]=0$$

$$D[v]=\infty \ (v \neq s)$$

として始め, 毎回 $D[v]$ の値が最小の頂点を選んで, その値を確定した後, 頂点 v に隣接する頂点への辺を調べて, 隣接頂点へのより短い経路を調べる.

Dijkstra's Algorithm

Find a shortest path from a starting vertex s to every other vertex in addition to a shortest one from s to a terminating vertex t .

$D[v]$: length of a shortest path from s to a vertex v .

We want to calculate the values $D[]$ for all the vertices.

Greedy choice:

Initially we have

$$D[s]=0$$

$$D[v]=\infty \ (v \neq s).$$

Every time we choose a vertex v

to minimize the value $D[v]$ and determine the value $D[v]$.

Then, we examine its adjacent edges to find shorter paths to its adjacent vertices.

アルゴリズムP14-A0:(ダイクストラ法)

$C = s$ 以外のすべての頂点からなる集合;

for すべての頂点 v do $D[v] = \infty$;

$D[s] = 0$;

$u = s$;

do{

 for 頂点 u に隣接するすべての頂点 v do

 if $D[u] + w(u, v) < D[v]$ then $D[v] = D[u] + w(u, v)$;

C の中で $D[]$ の値が最小の頂点を選び, u とする.

C から u を削除する.

} while(C が空でない);

演習問題: ダイクストラのアルゴリズムの計算時間について
他のテキストを調査せよ.

演習問題: アッカーマン関数について調査せよ.

Algorithm P14-A0: (Dijkstra's Algorithm)

```
C = a set of vertices except s;  
for each vertex v do  $D[v] = \infty$ ;  
 $D[s] = 0$ ;  
u = s;  
do {  
    for each vertex v adjacent to vertex u do  
        if  $D[u] + w(u, v) < D[v]$  then  $D[v] = D[u] + w(u, v)$ ;  
    Choose a vertex u to minimize  $D[]$  among C;  
    Remove u from C;  
} while (C is not empty);
```

Exercise: Examine textbooks for the computation time of the Dijkstra's algorithm.

Exercise: Examine the Ackermann function.

アルゴリズム論 Theory of Algorithms

第5回講義 枝刈り探索法

アルゴリズム論 Theory of Algorithms

Lecture #5 Prune and Search

枝刈り探索法の原理

与えられた問題を解くのに、構成要素を1つずつ吟味し、問題の解決に不必要だと思われるものを冗長なものとして削除し、縮小された問題を再帰的に解く。

サイズ n の問題を解くのに要する計算時間を $T(n)$ とする。
サイズ n の問題の構成要素を吟味して、サイズを rn ($r < 1$)の問題に縮小できるものとする。

構成要素の吟味が線形の cn 時間でできるとすると、漸化式

$$T(n) \leq T(rn) + cn$$

を得る。

この漸化式は次のように解ける。ただし、 $r^k n \leq 1$ とし、 $T(1)=d$ とする。

$$\begin{aligned} T(n) &\leq T(rn) + cn \leq T(r^2n) + crn + cn \leq T(r^3n) + cr^2n + crn + cn \\ &\leq T(r^kn) + cr^{k-1}n + \cdots + cr^2n + crn + cn \leq d + cn/(1-r). \end{aligned}$$

よって、 $T(n) = O(n)$ となり、線形時間であることがわかる。

Principle of Prune-and-Search

Solve a given problem by removing as many redundant elements as possible by examining every element and solving the reduced problem recursively.

$T(n)$: time to solve a problem of size n .

Assume that examination of every element of a problem of size n reduces the problem size into rn ($r < 1$).

If we can examine all the elements in cn time, then we have the recurrence equation

$$T(n) \leq T(rn) + cn.$$

The equation can be solved as follows: Here assume that $r^k n \leq 1$ and $T(1) = d$, a constant. Then, we have

$$\begin{aligned} T(n) &\leq T(rn) + cn \leq T(r^2n) + crn + cn \leq T(r^3n) + cr^2n + crn + cn \\ &\leq T(r^k n) + cr^{k-1}n + \dots + cr^2n + crn + cn \leq d + cn/(1-r). \end{aligned}$$

Thus, we have $T(n) = O(n)$, i.e., it runs in linear time.

定理: 問題のサイズに比例する時間で毎回一定の比率 r で問題のサイズを縮小することができるとき, この問題は枝刈り探索法により線形時間で解くことができる.

構成要素を吟味するのに $O(n)$ 時間ではなく $O(n \log n)$ 時間かかる場合の漸化式は

$$T(n) \leq T(rn) + cn \log n$$

となる. これを解くと,

$$T(n) \leq d + cn \log n / (1-r)$$

となるので, やはり $O(n \log n)$ 時間で問題が解ける.

演習問題: では, 構成要素の吟味に $O(n^2)$ 時間かかる場合はどうか?

レポート課題の問題3と似てますが, 違います ($r < 1$ なので).
おまけ問題: これもやってください.

Theorem: If we can reduce the problem size at some constant rate r for any problem instance, this problem can be solved in linear time based on a Prune-and-Search algorithm.

If it takes $O(n \log n)$ time to examine all the elements instead of $O(n)$ time, then the recurrence equation becomes

$$T(n) \leq T(rn) + cn \log n.$$

A solution to this equation is

$$T(n) \leq d + cn \log n / (1-r),$$

which implies that the problem is solved in $O(n \log n)$ time.

Exercise: What about the running time if it takes $O(n^2)$ time to investigate all the elements?

枝刈り探索法で解ける問題

一つの問題は以前に調べた中央値選択の問題.

アルゴリズムP9-A2:

- (1) n 個のデータを15個ずつのグループに分割し, 各グループごとに15個のデータの中央値を求める.
- (2) このようにして得られた $n/15$ 個の中央値に, この方法を再帰的に適用し, 中央値の中央値 M を求める.
- (3) n 個のデータを M に関して分割:
 S = M より小さいデータの集合,
 L = M より大きいデータの集合,
 E = M に等しいデータの集合,
- (4) $k \leq |L|$ のとき, L の中で k 番目に大きな要素を再帰的に求める.
- (5) $k > |L| + |E|$ のとき, S の中で $k - |L| - 1$ 番目に大きな要素を再帰的に求める.
- (6) それ以外の場合, M が求める答である.

Problems solved by Prune-and-Search

One of them is the median finding problem studied before.

Algorithm P9-A2:

- (1) decompose n data into groups each containing at most 15 data and find the median in each group.
- (2) Find the median M of these $n/15$ medians obtained recursively.
- (3) Decompose the n data with respect to M :
 - S = a set of data $< M$,
 - L = a set of data $> M$,
 - E = a set of data $= M$.
- (4) if $k \leq |L|$, find the k -th largest element in L recursively.
- (5) If $k > |L| + |E|$, find the $(k - |L| - 1)$ -th largest element in S recursively.
- (6) Otherwise return M as a solution.

このアルゴリズムでは,

- (1)全体を15個ずつの $(n/15)$ 個のグループに分け,
- (2)それぞれのグループ内でソートした後,
- (3)各グループの中央値を集めて, その中央値を求め,
- (4)その中央値に関して全体を分割し, 求める k 番目に
大きい要素がどのグループに属するかを求め,
- (5)そのグループにアルゴリズムを再帰的に適用した.

(1)の処理は定数時間(特に何もしない)

(2)の処理は $42 \times (n/15)$ に比例する時間

(3)の処理は $T(n/15)$ 時間.

(4,5)の処理は, 高々 $T((11/15)n)$ 時間. なぜなら, 分割によって
生じる集合のサイズはいずれも高々 $(11/15)n$ だから.

漸化式は

$$T(n) \leq 42(n/15) + T(n/15) + n + T((11/15)n)$$

となる. これを解くと,

$$T(n) \leq 19n.$$

This algorithm

- (1) decomposes the whole into $(n/15)$ groups of size at most 15,
- (2) sorts the elements in each group,
- (3) finds the median of the medians of all groups,
- (4) decomposes the whole by the median to find to which group the desired k -th largest element belongs to, and
- (5) applies the algorithm recursively to the group.

(1) is done in constant time (nothing in particular).

(2) is done in time proportional to $42 \times (n/15)$.

(3) is done in $T(n/15)$ time.

(4,5) are done in at most $T((11/15)n)$ time, because the size of each set resulting by the decomposition is at most $(11/15)n$.

The recurrence equation is given by

$$T(n) \leq 42(n/15) + T(n/15) + n + T((11/15)n).$$

Solving this, we have

$$T(n) \leq 19n.$$

2つのパラメータで指定される枝刈り探索

- ・構成要素を吟味する時間は問題のサイズ n に線形.
- ・吟味の後, サイズ an に縮小された問題と, サイズ bn に縮小された問題を再帰的に解くことにより, 元の問題を解決する.
- ・漸化式は

$$T(n) \leq T(an) + T(bn) + cn.$$

$0 < a + b < 1, 0 < a < 1, 0 < b < 1$ のとき, 上記の漸化式の解は

$$T(n) = O(n)$$

となる.

演習問題: 上記の観察を証明せよ.

Prune-and-Search using two parameters

- Time to examine all the elements is linear in the problem size.
- After the examination, we solve the original problem by solving the reduced problems of sizes an and bn recursively.
- The recurrence equation is

$$T(n) \leq T(an) + T(bn) + cn.$$

If $0 < a + b < 1$, $0 < a < 1$, $0 < b < 1$, the solution to the above equation is given by

$$T(n) = O(n).$$

Exercise: Prove the above observation.

2変数線形計画問題

問題P15:(2変数線形計画問題)

変数 x, y に関する多数の線形不等式をすべて満たす (x, y) の中で, 線形の目的関数 $cx+dy$ の値を最小(または最大)にするものを求めよ.

一般的な形式:

目的関数: $c_1x_1 + c_2x_2 \rightarrow \min$

制約条件: $d_{11}x_1 + d_{12}x_2 \geq e_1,$

$d_{21}x_1 + d_{22}x_2 \geq e_2,$

.....

$d_{n1}x_1 + d_{n2}x_2 \geq e_n.$

$y=c_1x_1 + c_2x_2, x=x_1$ とおくと,
 n 個の制約式は, d_{i2}/c_2 の
符号により

$$y \geq a_i x + b_i,$$

の形の下方制約式と

$$y \leq a_i x + b_i,$$

の形の上方制約式に分類
できる.

2-dimensional Linear Program

Problem P15: (2-dimensional Linear Program)

Find (x, y) to minimize (or maximize) a linear objective function $cx+dy$ among those satisfy all the linear inequalities on variables x and y .

General form:

Objective: $c_1x_1 + c_2x_2 \rightarrow \min$
subject to : $d_{11}x_1 + d_{12}x_2 \geq e_1,$
 $d_{21}x_1 + d_{22}x_2 \geq e_2,$
.....
 $d_{n1}x_1 + d_{n2}x_2 \geq e_n.$

If we set $y=c_1x_1 + c_2x_2$, $x=x_1$,
 n inequalities are classified into
lower constraints

$$y \geq a_i x + b_i,$$

and upper constraints

$$y \leq a_i x + b_i,$$

depending on signs of d_{i2}/c_2 .

例題:

目的関数: $x_1 + 2x_2 \rightarrow \min$

制約条件: (1) $-2x_2 \geq -5$, (2) $2x_2 \geq -3$, (3) $-3x_1 - 4x_2 \geq -14$,
(4) $x_1 + 4x_2 \geq 0$, (5) $2x_1 + 2x_2 \geq 3$, (6) $x_1 \geq -2$,
(7) $-x_1 \geq -8$.



$$y = x_1 + 2x_2, \quad x = x_1$$

目的関数: $y \rightarrow \min$

制約条件: (2') $y \geq x - 3$, (4') $y \geq (1/2)x$, (5') $y \geq -x + 3$,
(1') $y \leq x + 5$, (3') $y \leq -(1/2)x + 7$, (6'7') $-2 \leq x \leq 8$.

与えられた制約式をすべて満たす点 (x, y) のうち, y の値が最小なものを見つけることが問題.

点 $p=(x,y)$ がすべての制約式を満たすとき, 点 p は
実行可能(feasible)であると言い, そのような点の領域 F を
実行可能領域という.

Example:

Objective: $x_1 + 2x_2 \rightarrow \min$

subject to: (1) $-2x_2 \geq -5$, (2) $2x_2 \geq -3$, (3) $-3x_1 - 4x_2 \geq -14$,
(4) $x_1 + 4x_2 \geq 0$, (5) $2x_1 + 2x_2 \geq 3$, (6) $x_1 \geq -2$,
(7) $-x_1 \geq -8$.



$$y = x_1 + 2x_2, \quad x = x_1$$

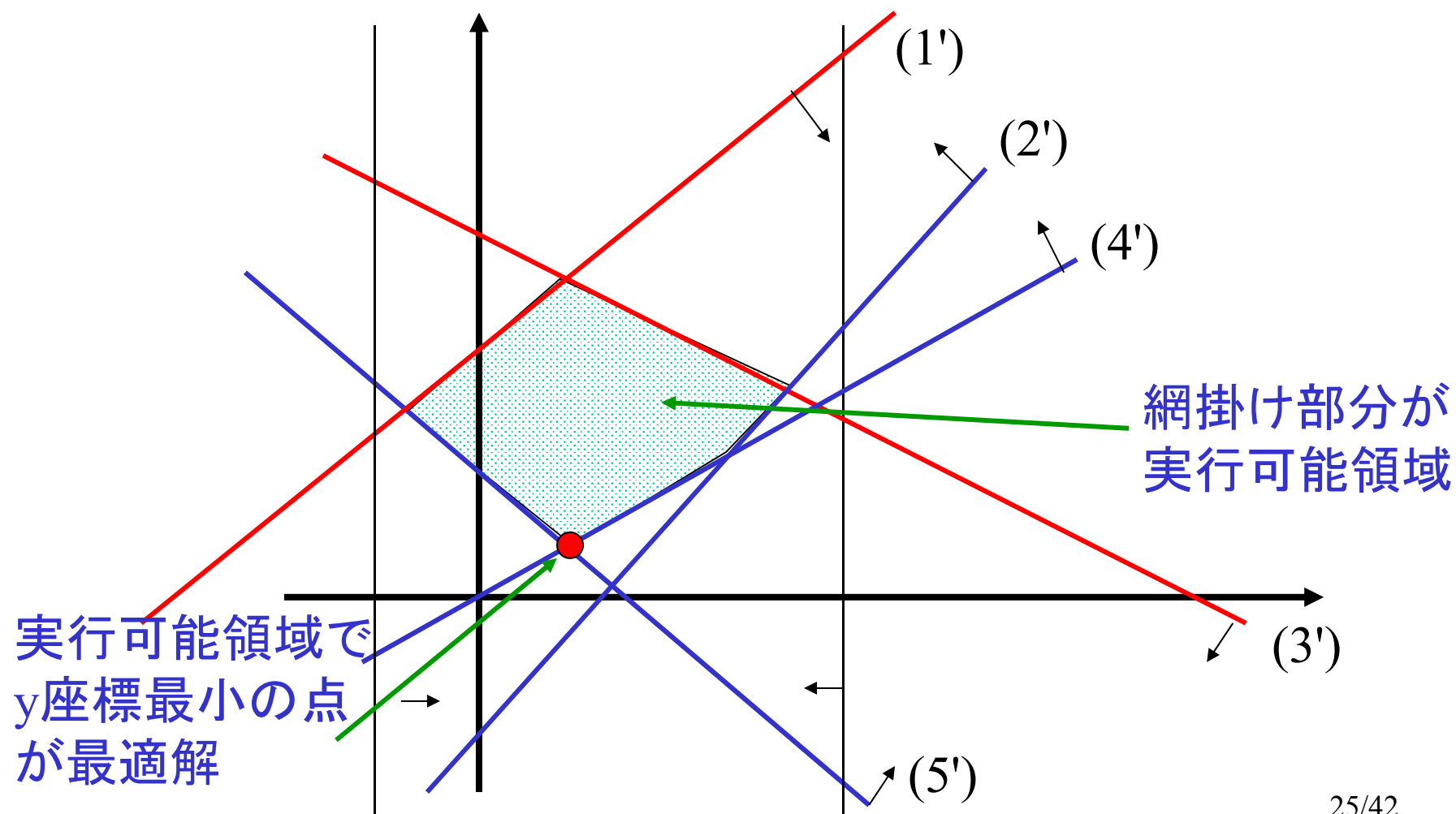
Objective: $y \rightarrow \min$

subject to: (2') $y \geq x - 3$, (4') $y \geq (1/2)x$, (5') $y \geq -x + 3$,
(1') $y \leq x + 5$, (3') $y \leq -(1/2)x + 7$, (6'7') $-2 \leq x \leq 8$.

Find a point (x, y) of the smallest y value among all those points satisfying all the constraints. When a point $p=(x,y)$ satisfies all the constraints, we say the point p is *feasible*, and a region of such feasible points is called a feasible region.

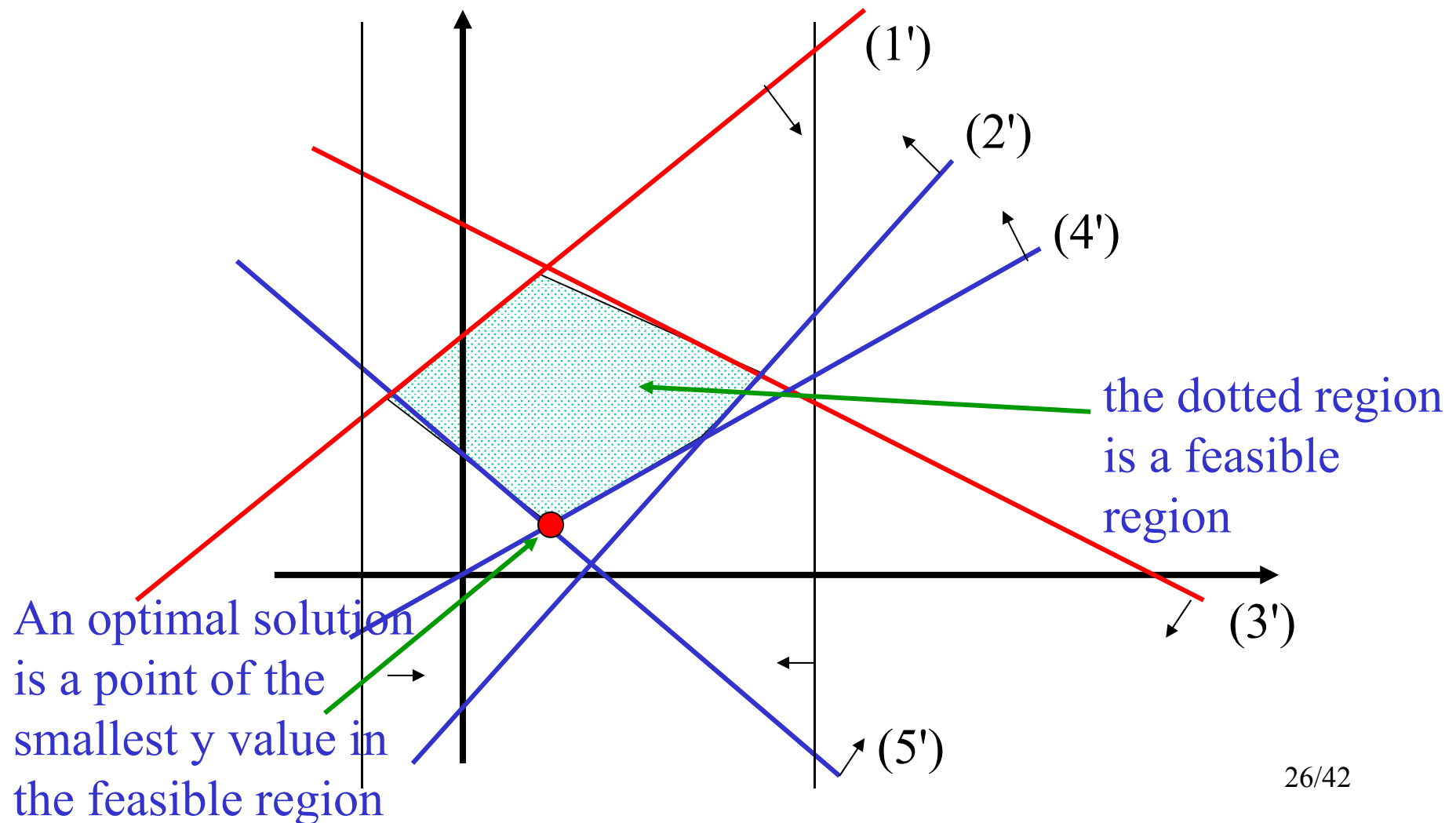
目的関数: $y \rightarrow \min$

制約条件: (2') $y \geq x-3$, (4') $y \geq (1/2)x$, (5') $y \geq -x+3$,
(1') $y \leq x+5$, (3') $y \leq -(1/2)x+7$, (6'7') $-2 \leq x \leq 8$.



Objective: $y \rightarrow \min$

subject to: (2') $y \geq x-3$, (4') $y \geq (1/2)x$, (5') $y \geq -x+3$,
(1') $y \leq x+5$, (3') $y \leq -(1/2)x+7$, (6'7') $-2 \leq x \leq 8$.



補題: 実行可能領域Fは, (もし存在すれば) 凸多角形を成す.

証明: 実行可能領域は制約式を満たす領域
下方制約式を満たす領域は下に凸,
上方制約式を満たす領域は上に凸,
よって, それらの凸領域の共通部分も凸である.

2つの関数の定義

$g(x)$: 下方制約式を満たす領域の境界を表す関数

$h(x)$: 上方制約式を満たす領域の境界を表す関数
すなわち,

$$g(x) = \max \{ a_i x + b_i \mid i \in I_1 \}$$

$$h(x) = \min \{ a_i x + b_i \mid i \in I_2 \}$$

と定めるとき, 実行可能領域Fは

$F = \{ (x, y) \mid g(x) \leq y \leq h(x) \text{ and } a \leq x \leq b \}$
と表現できる.

Lemma: A feasible region forms a convex polygon (if it exists).

Proof: A feasible region is a region satisfying all the constraints
The region satisfying the lower constraints is convex downward.
The region satisfying the upper constraints is convex upward.
Thus, the intersection of these convex regions is also convex.

Definition of two functions

$g(x)$: function representing the boundary of the region satisfying the lower constraints

$h(x)$: function representing the boundary for the upper constraints

That is, if we define

$$g(x) = \max \{a_i x + b_i \mid i \in I_1\}, \text{ and}$$

$$h(x) = \min \{a_i x + b_i \mid i \in I_2\}$$

then, the feasible region F is expressed by

$$F = \{(x, y) \mid g(x) \leq y \leq h(x) \text{ and } a \leq x \leq b\}.$$

2変数線形計画問題

目的関数: $y \rightarrow \min$

制約条件: $y \geq a_i x + b_i, i \in I_1$ (下方制約式)

$y \leq a_i x + b_i, i \in I_2$ (上方制約式)

$a \leq x \leq b,$

ただし, $|I_1| + |I_2| \leq n$ (制約式は高々 n 個)

アルゴリズムP15-A0: (分割統治法)

- ・上方制約式を2分割し, それぞれの部分に対応する実行可能領域(凸多角形)を再帰的に求めた後, ソート列をマージする要領で, 2つの凸多角形の共通部分を求める.
- ・同様に下方制約式についても分割統治法で実行可能領域を求める.
- ・2つの実行可能領域の共通部分を求め, その頂点で最も y 座標が小さいものを解として返す.

2-dimensional linear program

Objective: $y \rightarrow \min$

subject to: $y \geq a_i x + b_i, i \in I_1$ (lower constraints)

$y \leq a_i x + b_i, i \in I_2$ (upper constraints)

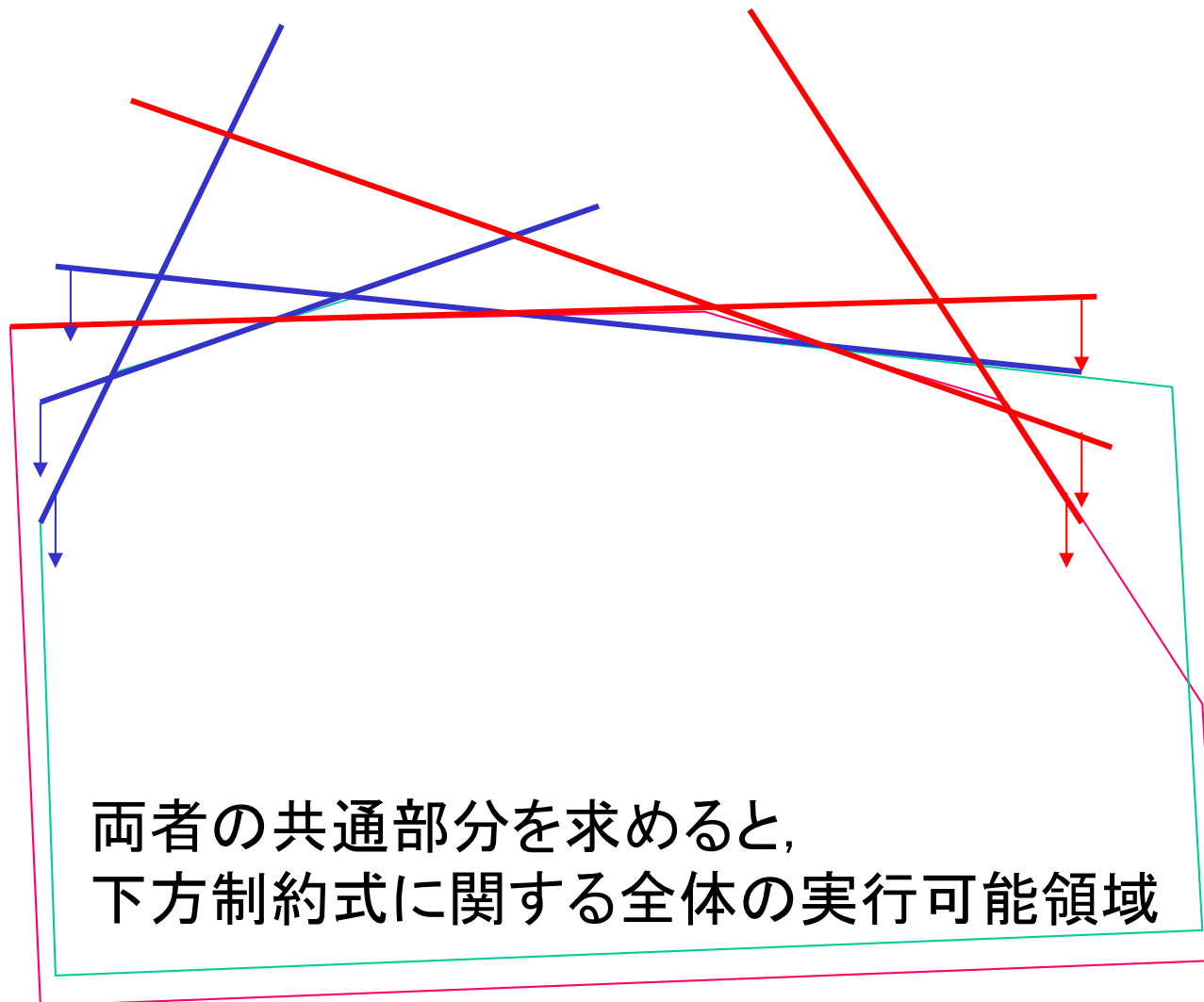
$a \leq x \leq b,$

Here, $|I_1| + |I_2| \leq n$ (there are at most n constraints.)

Algorithm P15-A0: (divide-and-conquer)

- Decompose the upper constraints into two parts, and find a feasible region (convex polygon) recursively for each part and then merge the two convex polygons to have their intersection.
- Similarly, compute a feasible region also for the lower constraints based on divide-and-conquer method.
- Find the intersection of the two feasible regions and return a vertex of the smallest y value as a solution.

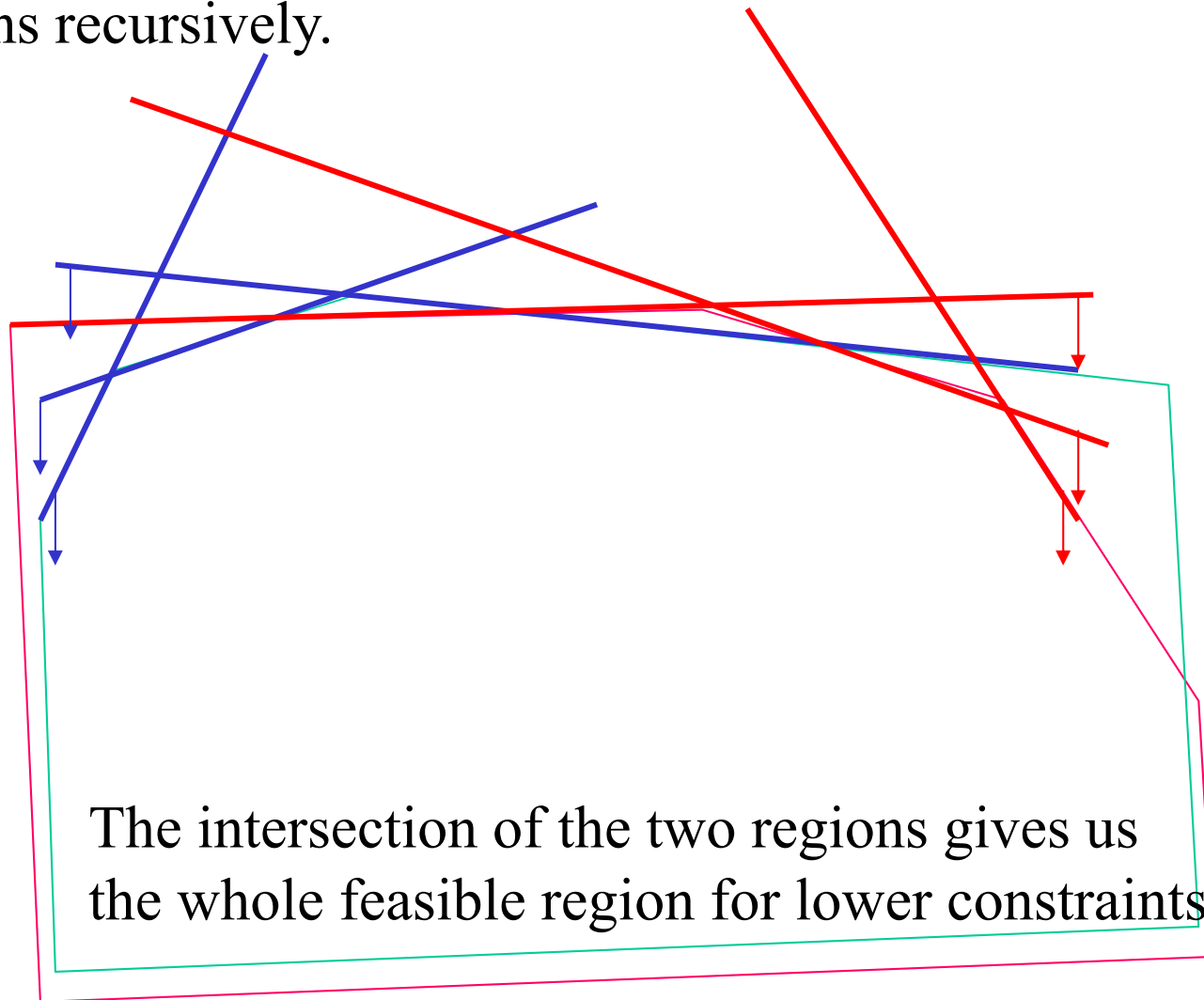
下方制約式の集合を2分割し, 再帰的に実行可能領域を求める



両者の共通部分を求めると,
下方制約式に関する全体の実行可能領域

演習問題: 分割統治法のアルゴリズムの計算時間は $O(n \log n)$ であることを示せ.

Divide the set of lower constraints into two and find feasible regions recursively.



Exercise: Prove that the algorithm based on divide-and-conquer runs in $O(n \log n)$ time.

n 個の制約式が与えられたとき, 対応する実行可能領域を陽に構成すれば, $O(n \log n)$ 時間かかる.

線形時間のアルゴリズムを得るには, 実行可能領域全体を陽に構成せずに, 最適解を与える頂点を求めることが必要.

枝刈り探索の考え方.

- ・制約式を吟味して, 一定割合の制約式を冗長な制約式として削除する.
- ・最適解を与える頂点は2つの制約式に対応する直線の交点として与えられる.
- ・関数 $g(x)$ と $h(x)$ の性質を利用する.
 $g(x)$ は下に凸, $h(x)$ は上に凸
- ・ただし, 関数 $g(x)$ と $h(x)$ の全体を求めようとすると, $O(n \log n)$ 時間かかってしまうことに注意.

Given n constraints, if we construct the corresponding feasible region in an explicit manner, then it takes $O(n \log n)$ time.

To obtain a linear-time algorithm, we need to find an optimal vertex without constructing the whole feasible region explicitly.

Idea based on Prune-and-Search

- By examining the constraints, we remove constraints by a fixed ratio as redundant constraints.
- A vertex giving an optimal solution is specified as an intersection of two lines corresponding to constraints.
- Use properties of the functions $g(x)$ and $h(x)$
 $g(x)$ is convex downward while $h(x)$ is convex upward.
- Note that if we compute the whole of the functions $g(x)$ and $h(x)$ then it takes $O(n \log n)$ time.

関数 $g(x)$ と $h(x)$ の性質を用いた枝刈り探索

- 性質(1) 下方制約に対応する関数 $g(x)$ は下に凸,
(2) 上方制約に対応する関数 $h(x)$ は上に凸,
(3) x の値の範囲は $[a, b]$.

区間 $[a, b]$ 内の任意の x' を指定したとき,
直線 $x=x'$ と下方制約式との交点の中で y 座標最大の点

$$g(x') = \max \{ a_i x' + b_i \mid i \in I_1 \}$$

直線 $x=x'$ と上方制約式との交点の中で y 座標最小の点

$$h(x') = \min \{ a_i x' + b_i \mid i \in I_2 \}$$

ここで, $g(x') \leq h(x')$ ならば, $x=x'$ は実行可能領域と交わる.
(注意) 上の関数値は $O(n)$ 時間で計算可能.

ケース1: $g(x') > h(x')$ のとき

ケース2: $g(x') \leq h(x')$ のとき

いずれの場合にも, 交点を与える直線の傾きにより冗長な制約式を削除する.

Prune-and-Search based on the properties of $g(x)$ and $h(x)$

Properties

- (1) The function $g(x)$ for the lower constraints is convex downward,
- (2) The function $h(x)$ for the upper constraints is convex upward, and
- (3) The range of x is $[a,b]$.

When we specify any x' in the interval $[a,b]$,

among intersections of the line $x=x'$ with lower constraints find the one of the largest y value gives $g(x')$

$$g(x') = \max \{a_i x' + b_i \mid i \in I_1\}$$

among intersections of the line $x=x'$ with upper constraints find the one of the smallest y value gives $h(x')$

$$h(x') = \min \{a_i x' + b_i \mid i \in I_2\}$$

If $g(x') \leq h(x')$ then the line $x=x'$ really intersect a feasible region.

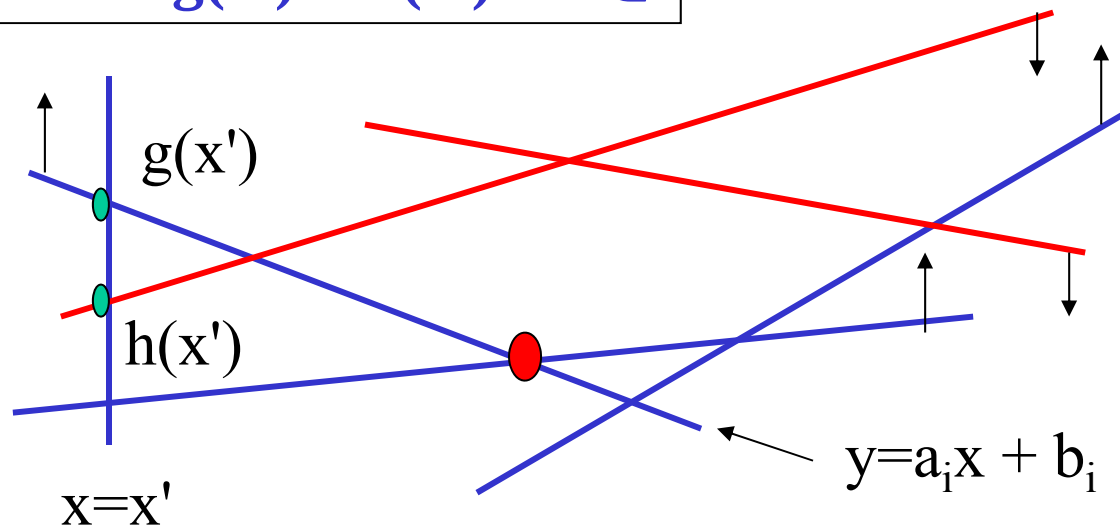
(Remark) The above function values are computed in $O(n)$ time.

Case 1: $g(x') > h(x')$

Case 2: $g(x') \leq h(x')$

In either case, we remove redundant constraints based on the slope of the line giving the intersection.

ケース1: $g(x') > h(x')$ のとき



$g(x')$ の値を与える制約式を $y \geq a_i x + b_i$ とする.

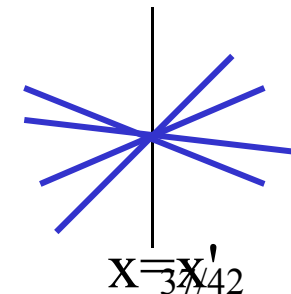
ケース1.1: $g(x')$ を与える制約式がただ1つの場合

この場合, $x=x'$ における関数 $y=g(x)$ の微分値 $g'(x')$ が $g'(x')=a_i$ として求まる.

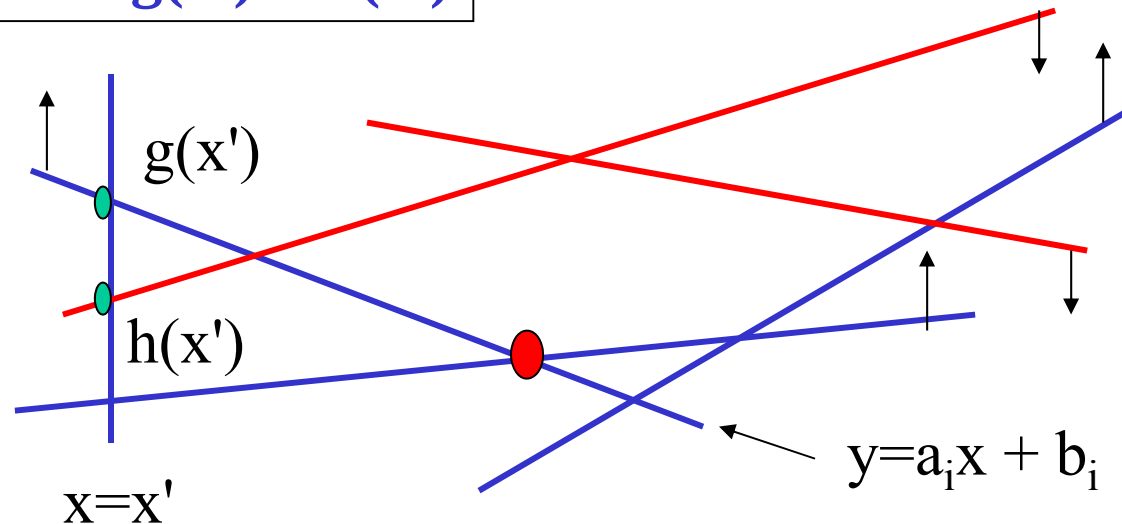
ケース1.2: $g(x')$ を与える制約式が複数個ある場合

それらの関数の中の最大の傾きが $x=x'$ の右での微分値

それらの関数の中の最小の傾きが $x=x'$ の左での微分値



Case 1 : $g(x') > h(x')$



Suppose the constraint $y \geq a_i x + b_i$ gives the value of $g(x')$.

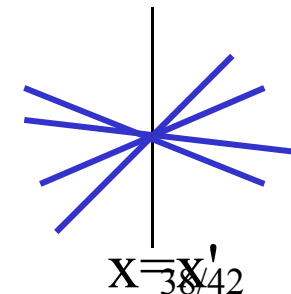
Case 1.1: Only one constraint gives $g(x')$

In this case, the differential value $g'(x')$ of the function $y=g(x)$ at $x=x'$ is obtained as $g'(x')=a_i$.

Case 1.2: More than one constraint give the value of $g(x')$

The largest slope among those functions is the differential value at the left of $x=x'$.

The smallest slope among those functions is the differential value at the right of $x=x'$.



$x=x'$

38/42

最適解がどちらの方向にあるかを知りたい

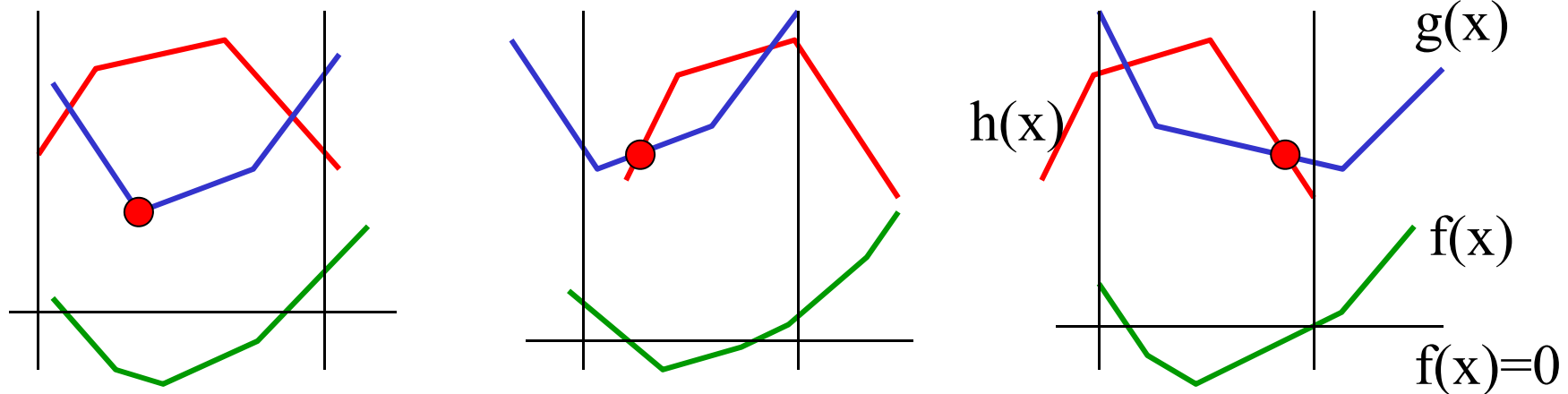
新たな関数 $f(x) = g(x) - h(x)$ の導入

$g(x)$: 下方制約式, 下に凸

$h(x)$: 上方制約式, 上に凸, $-h(x)$ は下に凸

$f(x)=g(x)-h(x)$ は下に凸.

(x,y) が実行可能 $\Leftrightarrow g(x) \leq y \leq h(x) \Leftrightarrow f(x) \leq 0$



$h(x') \leq g(x')$ が成り立つ様々な状況(縦線で示す)

$x=x'$ における $y=g(x)$ の傾きだけでは最適解の方向を
特定することはできない.

$y=f(x)$ の傾きを考えると, 最適解は $f(x)$ の値が減少する方向にある

Want to know in which side an optimal solution lies.

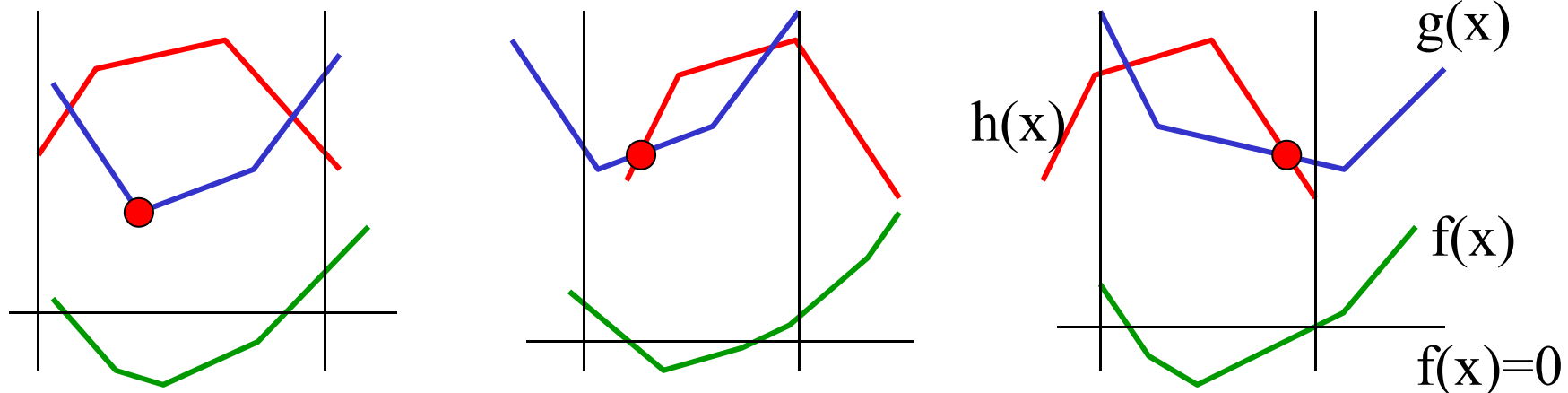
Introducing a new function $f(x) = g(x) - h(x)$.

$g(x)$: lower constraints, convex downward

$h(x)$: upper constraints, convex upward, $-h(x)$ is convex downward

$f(x) = g(x) - h(x)$ is convex downward.

(x, y) is feasible $\Leftrightarrow g(x) \leq y \leq h(x) \Leftrightarrow f(x) \leq 0$



Various situations (indicated by vertical lines) in $h(x') \leq g(x')$ holds

We cannot know the direction of an optimal solution using only slope of $y = g(x)$ at $x = x'$.

Considering the slope of $y = f(x)$, an optimal solution lies in the direction in which $f(x)$ value decreases.

関数 $y=f(x)$ の $x=x'$ における傾きを知りたい

$O(n)$ 時間で, $x=x'$ (の左右)における $y=g(x)$ の傾き

$O(n)$ 時間で, $x=x'$ (の左右)における $y=h(x)$ の傾き

これらを用いて $x=x'$ (の左右)における $y=f(x)$ の傾きを求める.

$f(x')>0$ かつ $x=x'$ の右における $y=f(x)$ の傾き <0 のとき

最適解は $x=x'$ の右にある.

$f(x')>0$ かつ $x=x'$ の左における $y=f(x)$ の傾き >0 のとき

最適解は $x=x'$ の左にある.

それ以外は

解は存在しない.

最適解の方向が分かると, 冗長な制約式を削除できる.

その方法は?

Want to know the slope of the function $y=f(x)$ at $x=x'$

$O(n)$ time suffices to find slopes of $y=g(x)$ at $x=x'$ (and around it).

$O(n)$ time suffices to find slopes of $y=h(x)$ at $x=x'$ (and around it).

Using them, find slopes of $y=f(x)$ at $x=x'$ (and around it).

When $f(x') > 0$ and the slope of $y=f(x)$ at the right of $x=x' < 0$
optimal solution lies to the right of $x=x'$.

When $f(x') > 0$ and the slope of $y=f(x)$ at the left of $x=x' > 0$
optimal solution lies to the left of $x=x'$.

Otherwise,

there is no solution.

Once the direction of an optimal solution is known, we can remove redundant constraints.

How?

$x=x'$ の右に最適解がある場合

下方制約式を2つずつペアにする.

上方制約式も2つずつペアにする.

制約式が奇数個なら, 最後の一つはペアにしない.



下方制約式のペア (i,j) $y \geq a_i x + b_i$, $y \geq a_j x + b_j$

交点を x_{ij} とする.

実行可能領域は $[a,b]$ の範囲にあった.

(1) $x_{ij} \leq a$ のとき,

最適解は右にある. よって, 傾きの小さい制約は冗長

(2) $x_{ij} \geq b$ のとき,

最適解は左にある. よって, 傾きの大きい制約は冗長

(3) それ以外するとき

When an optimal solution lies to the right of $x=x'$

Decompose lower constraints into pairs.

Decompose upper constraints into pairs.

If we have odd number of constraints, then leave the last one.



Pair (i, j) of lower constraints: $y \geq a_i x + b_i$, $y \geq a_j x + b_j$

Let their intersection be x_{ij} .

Note that the feasible region is in the region $[a, b]$.

(1) Case 1: $x_{ij} \leq a$,

Optimal solution lies to the right. So, the one of smaller slope in the pair is redundant.

(2) Case 2 $x_{ij} \geq b$,

Optimal solution lies to the left. So, the one of larger slope in the pair is redundant.

(3) Case 3: otherwise.

交点が $[a,b]$ の区間に入っているとき

そのようなペアについて交点 x_{ij} を求め、それらの中央値 x_m を求める.

$x=x_m$ において、 $g(x)$ と $h(x)$ の値を求めるなど、先と同じ計算を行う
このとき、下記のいずれかの結論を得る:

- (1)この問題に解はない.
- (2) x_m は最適解である.
- (3)最適解は $x=x_m$ の右にある.
- (4)最適解は $x=x_m$ の左にある.

(3)の場合、交点が $x=x_m$ の左にあるペア (i,j) については、
一方の制約式を冗長な制約式として削除可能.

(4)の場合、交点が $x=x_m$ の右にあるペア (i,j) については、
一方の制約式を冗長な制約式として削除可能.

したがって、約1/4の制約式を削除することができる.

=>枝刈り探索法により線形時間で最適解が求まる.

When the intersection lies in the interval $[a,b]$:

Find the intersection x_{ij} for each such pair and compute the median of those intersections.

Do the same calculations as before, say to compute $g(x)$ and $h(x)$ at $x=x_m$. Then, we have one of the followings as the result.

- (1) This problem has no solution.
- (2) x_m is the optimal solution.
- (3) An optimal solution lies to the right of $x=x_m$.
- (4) An optimal solution lies to the left of $x=x_m$.

For (3), we can remove one of the paired constraints whose intersection lies to the left of $x=x_m$ as a redundant constraint.

For (4), we can remove one of the paired constraints whose intersection lies to the right of $x=x_m$ as a redundant constraint.

Thus, we can remove about $\frac{1}{4}$ of all constraints.

➔ We can find an optimal solution in linear time using Prune-and-Search.

定理:2変数の線形計画問題は, 制約式の個数 n の線形時間で解ける.

証明:

簡単のため, どの制約式も x と y の両方に関係するものとする.

(1) n 個の制約式を下方制約式と上方制約式に分け, それぞれでペアを作る. それぞれのペアに対して対応する直線の交点を求める. そのような交点の個数はほぼ $n/2$ である.

(2)上記の交点たちの x 座標の中央値 x_m を求める.

(3)直線 $x=x_m$ と下方制約式に対応する下方境界と上方制約式に対応する上方境界との交点と, 交点を与える制約式を求めることにより, それらの交点における実行可能領域の傾きを求める.

(4)上記で求めた傾きにより最適解が直線 $x=x_m$ のどちら側にあるかを判定し, その反対側に交点をもつ制約式のペアについて一方の制約式を冗長な制約式として削除する. このとき削除される制約式は, 少なくとも $n/4$ 個ある.

Theorem: A 2-dimensional linear program can be solved in time linear in the number of constraints.

Proof:

For simplicity, we assume that every constraint is associated with both x and y .

(1) Divide the n constraints into lower and upper constraints and decompose them into pairs in each group. For each pair, compute the intersection of their associated lines. There are about $n/2$ such intersections in total.

(2) Find the median of those x -coordinates x_m .

(3) At the line $x=x_m$, we compute slopes of the boundary of the feasible region by finding the lower constraint giving the intersection with the lower boundary for lower constraints and the one for the upper boundary for upper constraints.

(4) Using the slopes obtained above, we check which side of the line $x=x_m$ contains an optimal solution, and then we remove as redundant constraints one of paired constraints having their corresponding intersection in the other side.

Then, there are at least $n/4$ constraints to be removed.

したがって、 n 個の制約式で定義される2変数線形計画問題を解くための時間を $T(n)$ とすると、

- (1)制約式のペアに対する交点は $O(n)$ 時間で計算できる.
- (2)も $O(n)$ 時間(最悪でも線形時間で中央値を求めることが可能).
- (3)境界との交点もすべての制約式との交点を調べればよいから $O(n)$ 時間で十分.
- (4)中央値の位置を利用して制約式を削除する操作も $O(n)$ 時間. これにより制約式は高々 $(3/4)n$ 個に減る.

したがって、 $O(n)$ の時間を適当な定数 c により cn と表せば、

$$T(n) \leq T((3/4)n) + cn$$

という漸化式を得る. これを解けば、 $T(n) = O(n)$ を得る.

演習問題: x だけの制約式や y だけの制約式があれば、それらはそれぞれどのように扱うべきか？

そのうちレポート課題にします

Therefore, if we denote by $T(n)$ the time to solve a 2-dimensional linear program, then we have

- (1) intersections for those pairs can be computed in $O(n)$ time in total,
- (2) is done in $O(n)$ time (median can be found in linear time in the worst case).
- (3) $O(n)$ time suffices to compute intersections with all constraints to compute the intersection with the boundary.
- (4) The operations to remove redundant constraints using the location of the median are done in $O(n)$ time. Then, the number of constraints is reduced to at most $(3/4)n$.

Hence, if we represent $O(n)$ time by cn for some constant c , then we have

$$T(n) \leq T((3/4)n) + cn,$$

from which we obtain $T(n) = O(n)$.

Exercise: How should we treat those constraints associated only with x or y ?