Design and Analysis of Algorithms (XII)

Simplex Algorithm

Guoqiang Li
School of Software

SHANGHAI JIAO TONG UNIVERSITY

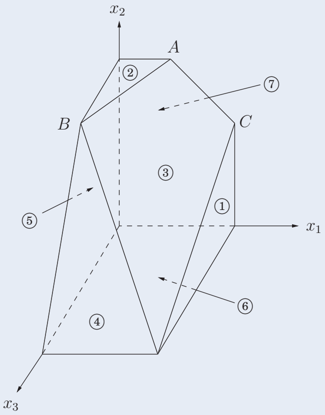
Review of Previous Lecture

LP

Shanghai Jiao Tong University

$$
\begin{aligned}
\max x_1 + 6x_2 + 13x_3 \\
x_1 \leq 200 \\
x_2 \leq 300 \\
x_1 + x_2 + x_3 \leq 400 \\
x_2 + 3x_3 \leq 600 \\
x_1, x_2, x_3 \geq 0
\end{aligned}
$$

The Example



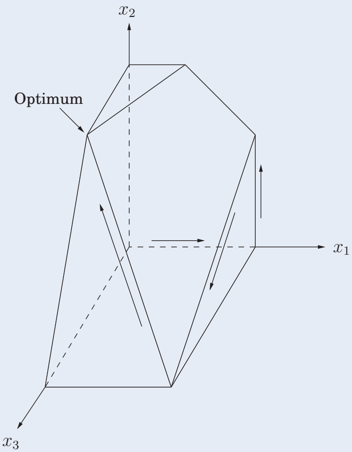
The point of final contact is the optimal vertex: $(0, 300, 100)$, with total profit $\$3100$.

Q: How would the simplex algorithm behave on this modified problem?

A possible trajectory

$$\frac{(0,0,0)}{\$0} \rightarrow \frac{(200,0,0)}{\$200} \rightarrow \frac{(200,200,0)}{\$1400} \rightarrow \frac{(200,0,200)}{\$2800} \rightarrow \frac{(0,300,100)}{\$3100}$$

The Example



LP and Its Dual

Shanghai Jiao Tong University

Primal LP	Dual LP
$\max c^T \mathbf{x}$	$\min \mathbf{y}^T b$
$A\mathbf{x} \le b$	$\mathbf{y}^T A \ge c^T$
$\mathbf{x} \ge 0$	$\mathbf{y} \ge 0$

Primal LP:

$$
\begin{aligned}
\max\ & c_1x_1 + \cdots + c_nx_n \\
a_{i1}x_1 + \cdots + a_{in}x_n &\le b_i \quad \text{for } i \in I \\
a_{i1}x_1 + \cdots + a_{in}x_n &= b_i \quad \text{for } i \in E \\
x_j &\ge 0 \quad \text{for } j \in N
\end{aligned}
$$

Dual LP:

$$
\begin{aligned}
\min\ & b_1y_1 + \cdots + b_my_m \\
a_{1j}y_1 + \cdots + a_{mj}y_m &\ge c_j \quad \text{for } j \in N \\
a_{1j}y_1 + \cdots + a_{mj}y_m &= c_j \quad \text{for } j \notin N \\
y_i &\ge 0 \quad \text{for } i \in I
\end{aligned}
$$

LP and Its Dual

$$\max x_1 + 6x_2$$
$$x_1 \le 200$$
$$x_2 \le 300$$
$$x_1 + x_2 \le 400$$
$$x_1, x_2 \ge 0$$

$$\min 200y_1 + 300y_2 + 400y_3$$
$$y_1 + y_3 \ge 1$$
$$y_2 + y_3 \ge 6$$
$$y_1, y_2, y_3 \ge 0$$

Complementary Slackness

SHANGHAI JIAO TONG UNIVERSITY

Theorem

Assume LP problem (P) has a solution x^* and its dual problem (D) has a solution y^*.

1. *If $x_j^* > 0$, then the j-th constraint in (D) is binding.*
2. *If the j-th constraint in (D) is not binding, then $x_j^* = 0$.*
3. *If $y_i^* > 0$, then the i-th constraint in (P) is binding.*
4. *If the i-th constraint in (P) is not binding, then $y_i^* = 0$.*

Standard Linear Programming

Variants of Linear Programming

A general linear program has many degrees of freedom:

Variants of Linear Programming

A general linear program has many degrees of freedom:

1. It can be either a maximization or a minimization problem.

Variants of Linear Programming

A general linear program has many degrees of freedom:

1. It can be either a maximization or a minimization problem.
2. Its constraints can be equations and/or inequalities.

Variants of Linear Programming

A general linear program has many degrees of freedom:

1. It can be either a maximization or a minimization problem.
2. Its constraints can be equations and/or inequalities.
3. The variables are often restricted to be nonnegative, but they can also be unrestricted in sign.

Variants of Linear Programming

A general linear program has many degrees of freedom:

1. It can be either a maximization or a minimization problem.
2. Its constraints can be equations and/or inequalities.
3. The variables are often restricted to be nonnegative, but they can also be unrestricted in sign.

We will now show that these various LP options can all be reduced to one another via simple transformations.

Variants of Linear Programming

SHANGHAI JIAO TONG UNIVERSITY

Variants of Linear Programming

To turn a maximization problem into a minimization (or vice versa), multiply the coefficients of the objective function by -1.

Variants of Linear Programming

To turn an inequality constraint like $\sum_{i=1}^{n} a_i x_i \leq b$ into an equation, introduce a new variable s and use

$$\sum_{i=1}^{n} a_i x_i + s = b$$

$$s \geq 0$$

This s is called the slack variable for the inequality.

Variants of Linear Programming

To turn an inequality constraint like $\sum_{i=1}^{n} a_i x_i \leq b$ into an equation, introduce a new variable s and use

$$\sum_{i=1}^{n} a_i x_i + s = b$$

$$s \geq 0$$

This s is called the slack variable for the inequality.

To change an equality constraint into inequalities is easy: rewrite $ax = b$ as the equivalent pair of constraints $ax \leq b$ and $ax \geq b$.

Variants of Linear Programming

SHANGHAI JIAO TONG UNIVERSITY

Variants of Linear Programming

Finally, to deal with a variable x that is unrestricted in sign, do the following:

- Introduce two nonnegative variables, $x^+, x^- \geq 0$.

Variants of Linear Programming

Finally, to deal with a variable x that is unrestricted in sign, do the following:

- Introduce two nonnegative variables, $x^+, x^- \geq 0$.
- Replace x, wherever it occurs in the constraints or the objective function, by $x^+ - x^-$.

Standard Form

We can reduce any LP into an LP of a much more constrained kind that we call the standard form:

Standard Form

Shanghai Jiao Tong University

We can reduce any LP into an LP of a much more constrained kind that we call the standard form:

- the variables are all nonnegative.
- the constraints are all equations.
- and the objective function is to be minimized.

Standard Form

We can reduce any LP into an LP of a much more constrained kind that we call the standard form:

- the variables are all nonnegative.
- the constraints are all equations.
- and the objective function is to be minimized.

$$\begin{aligned} \max x_1 + 6x_2 \\ x_1 \leq 200 \\ x_2 \leq 300 \\ x_1 + x_2 \leq 400 \\ x_1, x_2 \geq 0 \end{aligned} \quad \Longrightarrow \quad \begin{aligned} \min -x_1 - 6x_2 \\ x_1 + s_1 = 200 \\ x_2 + s_2 = 300 \\ x_1 + x_2 + s_3 = 400 \\ x_1, x_2, s_1, s_2, s_3 \geq 0 \end{aligned}$$

The Simplex Algorithm

General Description

Simplex

let v be any *vertex* of the feasible region, while there is a *neighbor* v' of v with better objective value: set $v = v'$

Vertices and Neighbors

SHANGHAI JIAO TONG UNIVERSITY

Definition (Vertex)

Each vertex is the unique point at which some subset of hyperplanes meet.

Vertices and Neighbors

Definition (Vertex)

Each vertex is the unique point at which some subset of hyperplanes meet.

Pick a subset of the inequalities. If there is a unique point that satisfies them with equality, and this point happens to be feasible, then it is a vertex.

Vertices and Neighbors

Definition (Vertex)

Each vertex is the unique point at which some subset of hyperplanes meet.

Pick a subset of the inequalities. If there is a unique point that satisfies them with equality, and this point happens to be feasible, then it is a vertex.

Each vertex is specified by a set of n inequalities (say there are n variables).

Vertices and Neighbors

Definition (Vertex)

Each vertex is the unique point at which some subset of hyperplanes meet.

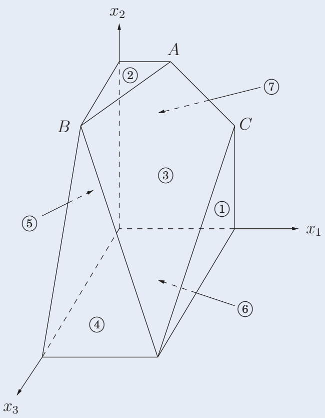
Pick a subset of the inequalities. If there is a unique point that satisfies them with equality, and this point happens to be feasible, then it is a vertex.

Each vertex is specified by a set of n inequalities (say there are n variables).

Definition (Neighbors)

Two vertices are neighbors if they have $n-1$ defining inequalities in common.

The Example



The Algorithm

SHANGHAI JIAO TONG UNIVERSITY

Algorithm

On each iteration, simplex has two tasks:

1. Check whether the current vertex is optimal (and if so, halt).

The Algorithm

Algorithm

On each iteration, simplex has two tasks:

1. Check whether the current vertex is optimal (and if so, halt).
2. Determine where to move next.

The Algorithm

Algorithm

On each iteration, simplex has two tasks:

1. Check whether the current vertex is optimal (and if so, halt).
2. Determine where to move next.

Both tasks are easy if the vertex is at the origin.

The Algorithm

SHANGHAI JIAO TONG UNIVERSITY

Algorithm

On each iteration, simplex has two tasks:

1. Check whether the current vertex is optimal (and if so, halt).
2. Determine where to move next.

Both tasks are easy if the vertex is at the origin. If the vertex is elsewhere, we transform the coordinate system to move it to the origin.

The Convenience for the Origin

Suppose we have some generic LP:

$$\max c^T \mathbf{x}$$
$$A\mathbf{x} \leq b$$
$$\mathbf{x} \geq 0$$

where $\mathbf{x}$ is the vector of variables, $\mathbf{x} = (x_1, \ldots, x_n)$.

The Convenience for the Origin

Suppose we have some generic LP:

$$\max c^T \mathbf{x}$$
$$A\mathbf{x} \le b$$
$$\mathbf{x} \ge 0$$

where $\mathbf{x}$ is the vector of variables, $\mathbf{x} = (x_1, \ldots, x_n)$.

Suppose the origin is feasible. Then it is certainly a vertex, since it is the unique point at which the n inequalities

$$\{x_1 \ge 0, \ldots, x_n \ge 0\}$$

are tight.

Task 1 in the Origin

Lemma

The origin is optimal if and only if all $c_i \leq 0$.

Task 1 in the Origin

SHANGHAI JIAO TONG UNIVERSITY

Lemma

The origin is optimal if and only if all $c_i \leq 0$.

Proof.

Task 1 in the Origin

SHANGHAI JIAO TONG UNIVERSITY

Lemma

The origin is optimal if and only if all $c_i \leq 0$.

Proof.

If all $c_i \leq 0$, then considering the constraints $x \geq 0$, we can't hope for a better objective value.

Task 1 in the Origin

Lemma

The origin is optimal if and only if all $c_i \leq 0$.

Proof.

If all $c_i \leq 0$, then considering the constraints $x \geq 0$, we can't hope for a better objective value.

Conversely, if some $c_i > 0$, then the origin is not optimal, since we can increase the objective function by raising x_i.

Task 2 in the Origin

Task 2 in the Origin

We can move by increasing some x_i for which $c_i > 0$.

Task 2 in the Origin

We can move by increasing some x_i for which $c_i > 0$.

Q: How much can we increase it?

Task 2 in the Origin

We can move by increasing some x_i for which $c_i > 0$.

Q: How much can we increase it?

A: Until we hit some other constraint.

Task 2 in the Origin

We can move by increasing some x_i for which $c_i > 0$.

Q: How much can we increase it?

A: Until we hit some other constraint.

We release the tight constraint $x_i \geq 0$ and increase x_i until some other inequality, previously loose, now becomes tight.

Task 2 in the Origin

We can move by increasing some x_i for which $c_i > 0$.

Q: How much can we increase it?

A: Until we hit some other constraint.

We release the tight constraint $x_i \geq 0$ and increase x_i until some other inequality, previously loose, now becomes tight.

We have exactly n tight inequalities, so we are at a new vertex.

An Example

$$\max 2x_1 + 5x_2$$

$$\begin{aligned} 2x_1 - x_2 &\leq 4 \\ x_1 + 2x_2 &\leq 9 \\ -x_1 + x_2 &\leq 3 \\ x_1 &\geq 0 \\ x_2 &\geq 0 \end{aligned}$$

Vertex Elsewhere

The trick is to transform u into the origin, by shifting the coordinate system from the usual $(x_1, \ldots, x_n)$ to the "local view" from u.

Vertex Elsewhere

The trick is to transform u into the origin, by shifting the coordinate system from the usual $(x_1, \ldots, x_n)$ to the "local view" from u.

These local coordinates consist of distances $y_1, \ldots, y_n$ to the n hyperplanes (inequalities) that define and enclose u.

Vertex Elsewhere

SHANGHAI JIAO TONG UNIVERSITY

The trick is to transform u into the origin, by shifting the coordinate system from the usual $(x_1, \ldots, x_n)$ to the "local view" from u.

These local coordinates consist of distances $y_1, \ldots, y_n$ to the n hyperplanes (inequalities) that define and enclose u.

If one of these enclosing inequalities is $a_i \cdot x \leq b_i$, then the distance from a point x to that particular "wall" is

$$y_i = b_i - a_i \cdot x$$

Vertex Elsewhere

The trick is to transform u into the origin, by shifting the coordinate system from the usual $(x_1, \dots, x_n)$ to the "local view" from u.

These local coordinates consist of distances $y_1, \dots, y_n$ to the n hyperplanes (inequalities) that define and enclose u.

If one of these enclosing inequalities is $a_i \cdot x \leq b_i$, then the distance from a point x to that particular "wall" is

$$y_i = b_i - a_i \cdot x$$

The n equations of this type, one per wall, define the y_i's as linear functions of the x_i's, and this relationship can be inverted to express the x_i's as a linear function of the y_i's.

An Example

$$\max 2x_1 + 5x_2$$

$$2x_1 - x_2 \leq 4$$
$$x_1 + 2x_2 \leq 9$$
$$-x_1 + x_2 \leq 3$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

An Example

Shanghai Jiao Tong University

$$\max 2x_1 + 5x_2$$

$$2x_1 - x_2 \leq 4$$
$$x_1 + 2x_2 \leq 9$$
$$-x_1 + x_2 \leq 3$$
$$x_1 \geq 0$$
$$x_2 \geq 0$$

$$\max 15 + 7y_1 - 5y_2$$

$$y_1 + y_2 \leq 7$$
$$3y_1 - 2y_2 \leq 3$$
$$y_2 \geq 0$$
$$y_1 \geq 0$$
$$-y_1 + y_2 \leq 3$$

An Example

$$\max 15 + 7y_1 - 5y_2$$

$$y_1 + y_2 \leq 7$$
$$3y_1 - 2y_2 \leq 3$$
$$y_2 \geq 0$$
$$y_1 \geq 0$$
$$-y_1 + y_2 \leq 3$$

An Example

$$\max 15 + 7y_1 - 5y_2$$

$$y_1 + y_2 \leq 7$$
$$3y_1 - 2y_2 \leq 3$$
$$y_2 \geq 0$$
$$y_1 \geq 0$$
$$-y_1 + y_2 \leq 3$$

$$\max 22 - 7/3z_1 - 1/3z_2$$

$$-1/3z_1 + 5/3z_2 \leq 6$$
$$z_1 \geq 0$$
$$z_2 \geq 0$$
$$1/3z_1 - 2/3z_2 \leq 1$$
$$1/3z_1 + 1/3z_2 \leq 4$$

Rewriting the LP

We can rewrite the entire LP in terms of the $\mathbf{y}$'s.

Rewriting the LP

SHANGHAI JIAO TONG UNIVERSITY

We can rewrite the entire LP in terms of the $\mathbf{y}$'s.

This doesn't fundamentally change, but expresses it in a different coordinate frame.

Rewriting the LP

We can rewrite the entire LP in terms of the $\mathbf{y}$'s.

This doesn't fundamentally change, but expresses it in a different coordinate frame.

The revised "local" LP has the following three properties:

1. It includes the inequalities $\mathbf{y} \geq 0$, which are simply the transformed versions of the inequalities defining u.
2. u itself is the origin in $\mathbf{y}$-space.
3. The cost function becomes $\max c_u + \tilde{c}^T \mathbf{y}$, where c_u is the value of the objective function at u and $\tilde{c}$ is a transformed cost vector.

Loose End

The Starting Vertex

In a general LP, the origin might not be feasible and thus not a vertex.

The Starting Vertex

In a general LP, the origin might not be feasible and thus not a vertex.

It turns out that finding a starting vertex can be reduced to an LP and solved by simplex!

The Starting Vertex

In a general LP, the origin might not be feasible and thus not a vertex.

It turns out that finding a starting vertex can be reduced to an LP and solved by simplex!

Start with any linear program in standard form:

$$\min c^T \mathbf{x} \text{ such that } \mathbb{A}\mathbf{x} = b \text{ and } x \geq 0.$$

The Starting Vertex

In a general LP, the origin might not be feasible and thus not a vertex.

It turns out that finding a starting vertex can be reduced to an LP and solved by simplex!

Start with any linear program in standard form:

$$\min c^T \mathbf{x} \text{ such that } \mathbb{A}\mathbf{x} = b \text{ and } x \geq 0.$$

We make sure that the right-hand sides of the equations are all nonnegative: if $b_i < 0$, multiply both sides of the i-th equation by -1.

The Starting Vertex

Then we create a new LP as follows:

- Create m new artificial variables $z_1, \ldots, z_m \geq 0$, where m is the number of equations.

The Starting Vertex

Then we create a new LP as follows:

- Create m new artificial variables $z_1, \ldots, z_m \geq 0$, where m is the number of equations.
- Add z_i to the left-hand side of the i-th equation.

The Starting Vertex

Then we create a new LP as follows:

- Create m new artificial variables $z_1, \ldots, z_m \geq 0$, where m is the number of equations.
- Add z_i to the left-hand side of the i-th equation.
- Let the objective, to be minimized, be $z_1 + z_2 + \ldots + z_m$.

An Example

$$
\begin{aligned}
\min -x_1 - 6x_2 \\
x_1 + s_1 = 200 \\
x_2 + s_2 = 300 \\
x_1 + x_2 + x_3 = 400 \\
x_1, x_2, x_3 \geq 0
\end{aligned}
$$

An Example

Shanghai Jiao Tong University

$$\min -x_1 - 6x_2$$
$$x_1 + s_1 = 200$$
$$x_2 + s_2 = 300$$
$$x_1 + x_2 + x_3 = 400$$
$$x_1, x_2, x_3 \geq 0$$

$$\min z_1 + z_2 + z_3$$
$$x_1 + s_1 + z_1 = 200$$
$$x_2 + s_2 + z_2 = 300$$
$$x_1 + x_2 + x_3 + z_3 = 400$$
$$x_1, x_2, x_3 \geq 0$$
$$z_1, z_2, z_3 \geq 0$$

The Starting Vertex

For this new LP, it's easy to come up with a starting vertex, namely, the one with $z_i = b_i$ for all i and all other variables zero.

The Starting Vertex

For this new LP, it's easy to come up with a starting vertex, namely, the one with $z_i = b_i$ for all i and all other variables zero.

Therefore we can solve it by simplex, to obtain the optimum solution.

The Starting Vertex

For this new LP, it's easy to come up with a starting vertex, namely, the one with $z_i = b_i$ for all i and all other variables zero.

Therefore we can solve it by simplex, to obtain the optimum solution.

There are two cases:

The Starting Vertex

For this new LP, it's easy to come up with a starting vertex, namely, the one with $z_i = b_i$ for all i and all other variables zero.

Therefore we can solve it by simplex, to obtain the optimum solution.

There are two cases:

1. If the optimum value of $z_1 + \ldots + z_m$ is zero,then all z_i's obtained by simplex are zero, and hence from the optimum vertex of the new LP we get a starting feasible vertex of the original LP.

The Starting Vertex

For this new LP, it's easy to come up with a starting vertex, namely, the one with $z_i = b_i$ for all i and all other variables zero.

Therefore we can solve it by simplex, to obtain the optimum solution.

There are two cases:

1. If the optimum value of $z_1 + \ldots + z_m$ is zero,then all z_i's obtained by simplex are zero, and hence from the optimum vertex of the new LP we get a starting feasible vertex of the original LP.
2. If the optimum objective turns out to be positive: We tried to minimize the sum of the z_i's, but it cannot be zero. This means that the original linear program is infeasible.

Degeneracy

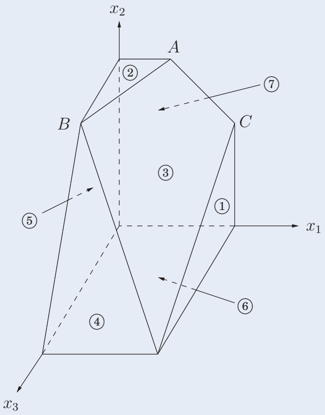
A vertex is degenerate if it is the intersection of more than n faces of the polyhedron, say $n+1$.

Degeneracy

A vertex is degenerate if it is the intersection of more than n faces of the polyhedron, say $n + 1$.

It means that if we choose any one of n sets of $n + 1$ inequalities and solve the corresponding system of these linear equations in n unknowns, we'll get the same solution in all $n + 1$ cases.

An Example



Degeneracy

This is a serious problem: simplex may return a suboptimal degenerate vertex simply because all its neighbors are identical to it and thus have no better objective.

Degeneracy

This is a serious problem: simplex may return a suboptimal degenerate vertex simply because all its neighbors are identical to it and thus have no better objective.

If we modify simplex so that it detects degeneracy and continues to hop from vertex to vertex despite lack of any improvement in the cost, it may end up looping forever.

Degeneracy

One way to fix this is by a perturbation:

Degeneracy

One way to fix this is by a perturbation:

change each b_i by a tiny random amount to $b_i \pm \varepsilon_i$.

Degeneracy

One way to fix this is by a perturbation:

change each b_i by a tiny random amount to $b_i \pm \varepsilon_i$.

This doesn't change the essence of the LP, but it has the effect of differentiating between the solutions of the linear systems.

Unboundedness

In some cases an LP is unbounded, in that its objective function can be made arbitrarily large (or small, if it's a minimization problem).

Unboundedness

In some cases an LP is unbounded, in that its objective function can be made arbitrarily large (or small, if it's a minimization problem).

If this is the case, simplex will discover it:

- In exploring the neighborhood, it will notice that taking out an inequality and adding another leads to an underdetermined system of equations.

Unboundedness

In some cases an LP is unbounded, in that its objective function can be made arbitrarily large (or small, if it's a minimization problem).

If this is the case, simplex will discover it:

- In exploring the neighborhood, it will notice that taking out an inequality and adding another leads to an underdetermined system of equations.
- The space of solutions contains a whole line across which the objective can become larger and larger, all the way to ∞.

Unboundedness

In some cases an LP is unbounded, in that its objective function can be made arbitrarily large (or small, if it's a minimization problem).

If this is the case, simplex will discover it:

- In exploring the neighborhood, it will notice that taking out an inequality and adding another leads to an underdetermined system of equations.
- The space of solutions contains a whole line across which the objective can become larger and larger, all the way to ∞.

In this case simplex halts and complains.

An Example

$$\max x_1 + x_2$$
$$x_1 - x_2 \geq 0$$
$$x_1, x_2 \geq 0$$

The Running Time

The Running Time of Simplex

Q: What is the running time of simplex, for a generic linear program:

$$\max c^T \mathbf{x} \text{ such that } \mathbb{A}\mathbf{x} \leq 0 \text{ and } \mathbf{x} \geq 0$$

where there are n variables and $\mathbb{A}$ contains m inequality constraints?

The Running Time of Simplex

Q: What is the running time of simplex, for a generic linear program:

$$\max c^T \mathbf{x} \text{ such that } \mathbb{A}\mathbf{x} \leq 0 \text{ and } \mathbf{x} \geq 0$$

where there are n variables and $\mathbb{A}$ contains m inequality constraints?

It is an iterative algorithm that proceeds from vertex to vertex. Let u be the current vertex.

The Running Time of Simplex

Q: What is the running time of simplex, for a generic linear program:

$$\max c^T \mathbf{x} \text{ such that } \mathbb{A}\mathbf{x} \leq 0 \text{ and } \mathbf{x} \geq 0$$

where there are n variables and $\mathbb{A}$ contains m inequality constraints?

It is an iterative algorithm that proceeds from vertex to vertex. Let u be the current vertex.

Each of its neighbors shares $n-1$ of these inequalities, so u can have at most $n \cdot m$ neighbors.

The Running Time of Simplex

A naive way for an iteration:

The Running Time of Simplex

A naive way for an iteration:

1. check each potential neighbor to see whether it really is a vertex of the polyhedron,

The Running Time of Simplex

A naive way for an iteration:

1. check each potential neighbor to see whether it really is a vertex of the polyhedron,
2. determine its cost.

The Running Time of Simplex

A naive way for an iteration:

1. check each potential neighbor to see whether it really is a vertex of the polyhedron,
2. determine its cost.

Finding the cost is quick, just a dot product.

The Running Time of Simplex

A naive way for an iteration:

1. check each potential neighbor to see whether it really is a vertex of the polyhedron,
2. determine its cost.

Finding the cost is quick, just a dot product.

Checking whether it is a true vertex involves: solve a system of n equations and check whether the result is feasible.

The Running Time of Simplex

A naive way for an iteration:

1. check each potential neighbor to see whether it really is a vertex of the polyhedron,
2. determine its cost.

Finding the cost is quick, just a dot product.

Checking whether it is a true vertex involves: solve a system of n equations and check whether the result is feasible.

By Gaussian elimination this takes $O(n^3)$ time, giving total $O(mn^4)$ per iteration.

The Running Time of Simplex

A much better way: the mn^4 can be improved to mn.

The Running Time of Simplex

SHANGHAI JIAO TONG UNIVERSITY

A much better way: the mn^4 can be improved to mn.

Recall the local view from vertex u. The per-iteration overhead of rewriting the LP in terms of the current local coordinates is just $O((m+n)n)$.

The Running Time of Simplex

A much better way: the mn^4 can be improved to mn.

Recall the local view from vertex u. The per-iteration overhead of rewriting the LP in terms of the current local coordinates is just $O((m+n)n)$.

The local view changes only slightly between iterations, in just one of its defining inequalities.

The Running Time of Simplex

To select the best neighbor, we recall that the objective function is of the form

$$\max c_u + \tilde{c} \cdot \mathbf{y}$$

where c_u is the value of the objective function at u.

The Running Time of Simplex

To select the best neighbor, we recall that the objective function is of the form

$$\max c_u + \tilde{c} \cdot \mathbf{y}$$

where c_u is the value of the objective function at u.

This immediately identifies a promising direction to move: we pick any $\tilde{c}_i > 0$.

The Running Time of Simplex

To select the best neighbor, we recall that the objective function is of the form

$$\max c_u + \tilde{c} \cdot \mathbf{y}$$

where c_u is the value of the objective function at u.

This immediately identifies a promising direction to move: we pick any $\tilde{c}_i > 0$.

Since the rest of the LP has now been rewritten in terms of the $\mathbf{y}$-coordinates, it is easy to determine how much y_i can be increased before some other inequality is violated.

The Running Time of Simplex

Q: How many iterations could there be?

The Running Time of Simplex

Q: How many iterations could there be?

A: At most $\binom{m+n}{n}$, i.e., the number of vertices.

The Running Time of Simplex

Shanghai Jiao Tong University

Q: How many iterations could there be?

A: At most $\binom{m+n}{n}$, i.e., the number of vertices.

It is exponential in n.

The Running Time of Simplex

Q: How many iterations could there be?

A: At most $\binom{m+n}{n}$, i.e., the number of vertices.

It is exponential in n.

And in fact, there are examples of LPs for which simplex does indeed take an exponential number of iterations.

The Running Time of Simplex

Q: How many iterations could there be?

A: At most $\binom{m+n}{n}$, i.e., the number of vertices.

It is exponential in n.

And in fact, there are examples of LPs for which simplex does indeed take an exponential number of iterations.

Simplex is an exponential-time algorithm.

The Running Time of Simplex

Q: How many iterations could there be?

A: At most $\binom{m+n}{n}$, i.e., the number of vertices.

It is exponential in n.

And in fact, there are examples of LPs for which simplex does indeed take an exponential number of iterations.

Simplex is an exponential-time algorithm.

However, such exponential examples do not occur in practice, and it is this fact that makes simplex so valuable and so widely used.

A Notable Result

Smoothed analysis proposed by Daniel Spielman and Shanghua Teng is a way of measuring the complexity of an algorithm. It gives a more realistic analysis of the practical performance of the algorithm. It was used to explain that the simplex algorithm runs in exponential-time in the worst-case and yet in practice it is a very efficient algorithm, which was one of the main motivations for developing smoothed analysis. The authors received the 2008 Gödel Prize and the 2009 Fulkerson Prize.

Referred Materials

Referred Materials

SHANGHAI JIAO TONG UNIVERSITY

Content of this lecture comes from Section 7.6 in [DPV07].