A Subexponential Randomized Simplex Algorithm

Gil Kalai (extended abstract)

Shimrit Shtern

Presentation for

Polynomial time algorithms for linear programming 097328

Technion - Israel Institute of Technology

May 14, 2012

Perliminaries

Linear Programming

Maximizing a linear function of d variables and n constraints over a the *feasable polyhedron.*

$$max\{c^T x : Ax \leq b\}$$

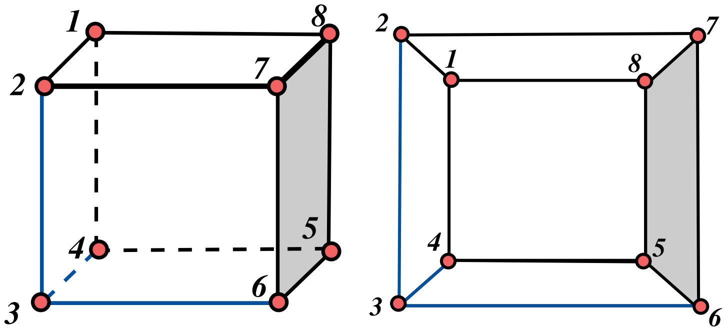
The dual

Minimizing a linear function of n variables and d constraints.

$$min\{b^T y : A^T y = c, y \geq 0\}$$

Linear programming properties

- The maximum (if exists) is achieved in one of the vertices.
- $P = \{x \in \Re^d : Ax \leq b\}$ the feasible polyhedron
- $G(P)$ is the a graph which connects vertices in P if they form the end points of a 1-dimensional face of P.



(a) Polygon - P

(b) Graph - G(P)

The Simplex Algorithm

- Introduction of the simplex [Dantzig, 1950]
- In each step we move from one feasible vertex v to an adjacent vertex w which has a higher objective function value.
- The choice of w is called the pivot rule.
- The simplex algorithm is a class of algorithms depending on the specific pivot rules.
- Number of pivot steps to the top.
- Common pivot rules proves to be exponential in worst case [Klee & Minty, 1971]
- Deterministic vs. Randomized pivot rules

Backround - Strong Algorithms

- A *Strong* algorithm - the number of arithmetic operations depends only on d and n (does not depend on L).
- The ellipsoid method and interior point method are polynomial but not strong.
- The existence of a strong polynomial algorithm for LP is still open (found for specific types of LP)
- Strong algorithms for LP which are linear for fixed dimensions: $O(2^{2^d}n)$ [Megiddo, 1984], $O(3^{d^2}n)$ [Dyer, 1986], [Clarkson, 1986], $O(d!n)$ [Seidel,1990]
- Clarkson (1988) - randomized algorithm, solves $O(d^2 logn)$ smaller linear programs with $O(d^2)$ constraints and d variables with expected $O(d^{\log\log n} + d^d) + O(d^2 n)$ operations.

A strong sub-exponential algorithm for LP

The Algorithm [Gil Kalai, 1992]

Randomized pivot rule with sub-exponential expected number of pivot steps

Used as a subroutine in Clarkson's Algorithm, we get a randomized dual-simplex algorithm which requires an expected $d^{O\left(\sqrt{d/\log d}+\log\log n\right)} + O(d^2 n)$ arithmetic operations.

Polyhedron Diameter Problem

- The longest shortest path between two vertices in a graph.
- Given an objective function, the longest shortest monotone path from a vertex to the top of the graph.
- An algorithm which finds a path from a vertex to the top of the graph.

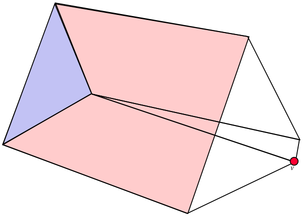
Definitions

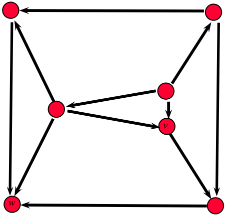
- $L(d, n)$ - class of LPs , d variables, n constraints .
- $O \in L(d, n)$ with objective function ϕ.
- P - the *feasible polyhedron* of O
- $G(P)$- the *feasible polyhedron graph*.
- Given $G(P) = (V, E)\,, \phi$ we define $\vec{G}(O) = (V, \tilde{E})$ - the *problem graph*:

$$(v, w) \in \tilde{E} \Leftrightarrow \{v, w\} \in E\,, \phi(w) > \phi(v)$$

- Extension:
  If ϕ is not bounded on P we add vertex $v = \infty$:
  $\phi(v) = \infty$ and (w, v) exists if there is a 1-dimensional face containing w on which ϕ is not bounded above.

Definitions (cont.)





- $\delta(G(P))$ - *diameter* of the graph $G(P)$.
- *Optimal set* -

$$W = \{w \in V : \phi(w) \geq \phi(v) \quad \forall v \in V\}$$

- $h(v)$ - the minimal length of a directed path from vertex v to some $w \in W$.
- $\delta(\vec{G}(O))$ - *height* of $\vec{G}(O)$

$$\delta(\vec{G}(O)) = \max_{v \in V} h(v)$$

Diameter and Height Bounds

- Diameter and Height of a polyhedra set:

$$\Delta(d,n) = \max_{O \in L(d,n)} \delta(G(P))$$
$$H(d,n) = \max_{O \in L(d,n)} \delta(\vec{G}(O))$$

- $\Delta(d,n) \leq H(d,n)$
- Hirsch conjuncture: $\Delta(d,n) \leq n-d$ **proven false for unbounded polyhedra** [Klee & Walkup, 1967]
- The best lower bound known :$\Delta(d,n) \geq n-d+\lfloor \frac{d}{5} \rfloor$
- Upper bound:
  - Exponential: $\Delta(d,n) \leq n2^{d-3}$ [Larman, 1970].
  - Quasi polynomial: $\Delta(d,n) \leq n^{2\log d+3}$ [Gil Kalai,1992], $n^{\log d+1}$ [Kalai & Kleitman,1992]

Properties

The objective

Finding a sub-exponential bound for for $H(d, n)$

$H(d, n)$ - minimal # of pivot steps when each step can have unlimited computational power.

More definitions

- $\mathcal{F}$ - *facets* of P ($d-1$ dimensional faces)
- $U(w) \subset \mathcal{F}$ the *active facets* of P with respect to w -

$$U(w) = \{F \in \mathcal{F} : max\{\phi(x) : x \in F\} > \phi(w)\}$$

- $u(w) = |U(w)|$
- $\bar{H}(d,n) = \max_{O \in L(d,\cdot)} \max_{v \in V : u(v) \leq n} h(v)$
- $H(d,n) \leq \bar{H}(d,n)$

A sub-exponential bound

Theorem

$$\bar{H}(d,n) \leq n \binom{d+\log n}{\log n} \leq n^{\log d+1}$$

We obtain this bound by finding a recursive formula for $\bar{H}(d,n)$.

A sub-exponential bound

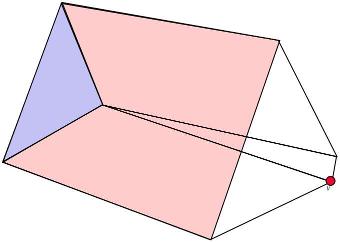
Proof outline

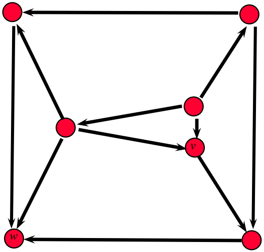
- We can reach $n-k+1$ active facets in at most $\bar{H}(d, n-k)$
- There exist an optimal vertex w in one of those facets $u(w) \leq k-1$
- We obtain the maximal vertex in that facet in $\bar{H}(d-1, n-1)$ steps.
- We continue from w with at most $\bar{H}(d, k-1)$ steps.

$$\bar{H}(d,n) \leq \bar{H}(d,n-k) + \bar{H}(d-1,n-1) + \bar{H}(d,k-1)$$

A sub-exponential bound - Proof

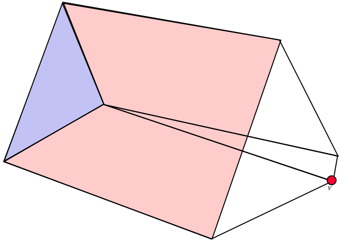
- $O \in L(d, \cdot)$,$\vec{G}(O) = (V, \tilde{E})$, $v \in V$ such that $u(v) \leq n$
- Let $S \subset U(v)$ and $|S| = k$
- B is the upper bound on the shortest monotone path from v to either a vertex in S or the top of P.

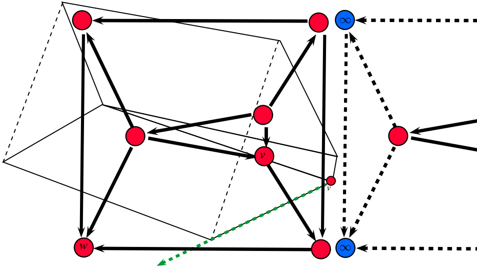




A sub-exponential bound - Proof (cont.)

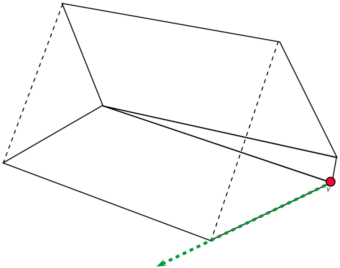
- case 1: $\exists F \in S : v \in F$ then $B = 0$
- case 2: $\forall F \in S : v \notin F$
  - Let O' the problem where $\mathcal{F}' = U(v) \bigcup \{F : v \in F\} \setminus S$ and P' the feasible polyhedron.
  - v is a vertex in the new problem.
  - In P' we have $u(v) \leq n - k \Rightarrow h(v) \leq \bar{H}(d, n - k)$

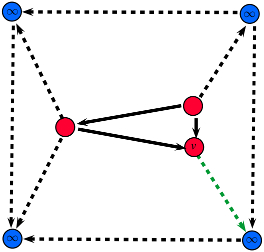




A sub-exponential bound - Proof (cont.)

- conclusion: $B \leq \bar{H}(d, n-k)$. why?
  - If the path which defines $h(v)$ is in P we have the latter inequality.
  - Otherwise, $\exists(x, y)$ in the path such that $x \in P$ and $y \notin P \Rightarrow \exists z \in S$ which is on the edge between x and y





A sub-exponential bound - Proof (cont.)

- Since this is true for every subgroup of size k we can reach at least $n-k+1$ facets in $\bar{H}(d, n-k)$ steps.
- On a certain facet it takes $\bar{H}(d-1, n-1)$ steps to get to the maximal vertex in that facet.
- The top vertex among those $n-k+1$ facets w satisfies $u(w) \leq k-1$
- We then get the recursive formula:

$$\bar{H}(d, n) \leq \bar{H}(d-1, n-1) + \bar{H}(d, n-k) + \bar{H}(d, k-1)$$

A sub-exponential bound - Proof (cont.)

- The recursive formula for $\bar{H}(d,n)$:

$$\bar{H}(d,n) \le \bar{H}(d-1,n-1) + \bar{H}(d,n-k) + \bar{H}(d,k-1)$$

- Setting $k = \lceil \frac{n}{2} \rceil$ and defining $f(d,t) = 2^t \bar{H}(d,2^t)$ we get:

$$\begin{aligned} f(d,t) &\le f(d-1,t) + f(d,t-1) \quad \Rightarrow \\ f(d,t) &\le \binom{d+t}{t} \quad \Rightarrow \\ \bar{H}(d,n) &\le n \binom{d+\log n}{\log n} \end{aligned}$$

Some remarks

- Bound on $\Delta(d, n)$ suggests every (primal) simplex algorithm needs at least a linear in n number of pivot steps.
- The upper bound is slightly super linear in n (for fixed d). **Is there a linear bound?**

Kalai's Algorithms

- Kalai presents three algorithms which are produced by 3 randomized pivot rules. (We will present two of them).
- The analysis of these algorithms is done by recursion.
- All the algorithms yield a sub-exponential expected number of pivot steps.
- Each pivot step takes at most $O(d^2 n)$ and generates at most 1 random variable

Assumptions and General Step

- P is a simple polyhedron , i.e. each vertex is the intersection of exactly d facets.
- We start from a feasible vertex v.
- We can "find" vertices on r active facets
  1. Start from $z := v$, $k := |S| = d$
  2. Solve an LP with group S known constraints (facets) recursively and obtain a solution z
  3. If z is in P then z is optimal.
  4. If z is not in P find the first edge E on the path which leaves the P. The last point on E in P is the intersection with facet $F \notin S$.
  5. $z := E \bigcap F$, $S := S \bigcup F$, $k := k + 1$
  6. If we have r facets, stop; otherwise go to step 2

Algorithm S_2

- Start from some vertex v:
  1. Find vertices in r active facets - $F_1, F_2, \ldots, F_r$
  2. Pick at random a facet from the r facets you reached.
  3. Find w the optimal vertex on that facet.
  4. Go back to step 1 beginning from w
- If we're going back to previously visited vertices why is this a simplex algorithm?
- What is $u(w)$?

Algorithm S_2 - Analysis

- Let $f_2(d,n)$ be the maximal expected number of pivot steps needed using algorithm S_2 for any problem in $L(d,n)$.
- The recursive formula for $f_2(d,n)$ implied from this algorithm is:

$$f_2(d,n) \leq \sum_{i=d}^{r} f_2(d,i) + f_2(d-1,n-1) + \frac{1}{r}\sum_{i=1}^{r} f_2(d,n-i)$$

- Choosing $r = \max\{d, \frac{n}{2}\}$ we get

$$f_2(d,n) \leq \sum_{i=d}^{\frac{n}{2}} f_2(d,i) + f_2(d-1,n-1) + \frac{2}{n}\sum_{i=1}^{\frac{n}{2}} f_2(d,n-i)$$

Algorithm S_2 - Analysis

Bounds

$$
\begin{aligned}
f_2(d,n) &\leq n^{O\left(\sqrt{\frac{d}{\log d}}\right)} \\
f_2(d,Kd) &\leq 2^{O(\sqrt{dK})} \\
f_2(d,d+m) &\leq 2^{O(\sqrt{m \log d})}
\end{aligned}
$$

Algorithm S_0

- Start from some vertex v:
  1. Pick at random a facet containing the current vertex.
  2. Find w the optimal vertex on that facet.
  3. Go back to step 1 beginning from w.
- What is $u(w)$?

Algorithm S_0 - Analysis

- Let $f_0(d,n)$ be the maximal expected number of pivot steps needed using algorithm S_0 for any problem in $L(d,n)$.
- The recursive formula for $f_0(d,n)$ implied from this algorithm is:

$$f_0(d,n) \leq f_0(d-1,n-1) + \frac{1}{d}\sum_{i=0}^{d-1} f_0(d,n-i)$$

Algorithm S_0 - Analysis

Bounds

$$f_0(d, d+m) \quad \leq 2^{O(\sqrt{m \log d})}$$

which is generally worse than the f_2 bound (specifically for $n = O(d^2)$)

Algorithm S_0 - Dual Form

- Independent work of Micha Sharir and Emo Welzl [1992] found a combinatorial randomized algorithm for Linear Programming
- The algorithm is a dual simplex algorithm, a variation of the dual form of S_0.
- The number of expected operation given this algorithm [J. Matoušk, Sharir and Welzl, 1992]:

$$min\{O(d^2 2^d n), e^{2\sqrt{d\ln(n/\sqrt{d})}+O(\sqrt{d}+\ln n)}\}$$

Conclusions

Outline

LP and Simplex

Polyhedron Diameter Problem

Formulation

Kalai's Theorem

Kalai's Algorithms

Algorithm S_2

Algorithm S_0

Summary

- Using oracle instead of randomization algorithm S_2 becomes quasi-polynomial (bound on $H(d, n)$).
- Using Kalai's algorithm as a subroutine in Clarksons algorithm we get an algorithm with expected arithmetic operations: $d^{O\left(\sqrt{d/\log d} + \log\log n\right)} + O(d^2 n)$.

Example:

- $d = 1,000,000$
- $n = 2d$ (implying $2^{O(\sqrt{d})}$ pivot steps)
- $10^{400,000}$ vertices.
- $10^{3,000}$ expected pivot steps using S_2.
- $H(d, n) \leq 10^{50}$

Open questions

- Is there a strong polynomial algorithm for linear programming?
- Decide if $\Delta(d, n)$ and $H(d, n)$ are polynomial.
- Decide if $H(d, n)$ is linear in n for a fixed dimension d.
- Deterministic sub-exponential pivot rules.
- Better randomized pivot rules.
- Give a randomized (primal) pivot rule for which for a fixed dimension the complexity is at most $O(n^C)$.

Questions?

Dual Simplex

- Tries to finds a subset of constraints S which contain optimal basis B.
- How?
  1. $S = \emptyset$
  2. Choose some R set of m constraints $m > d$.
  3. Use standard simplex algorithm to compute optimum over $R \bigcup S$ returns vertex v.
  4. Computes Z, the set of violated constraints. If Z is empty than v is optimal.
  5. Update $S = S \bigcup Z$ and return to step 2.
- At most d iterations to obtain a solution.
- determine m such that Z is not too large.

Clarkson's Algorithm

- In Clarkson's algorithm, R is random and of size $d\sqrt{n}$
- Insures an expected $\sqrt{n}$ constraints violated at each iteration.
- If there are less than $9d^2$ constraints, the optimum is found using a simple algorithm as a base subroutine.
- At most $O(d^2 logn)$ calls to the base subroutine.

Why is S_2 a simplex?

Lets look at the randomization in a slightly different way:

- Order the facets we encounter by the order we encounter them, e.g. F_1 is the first facet we encounter and F_r the last.
- Generate, in advance, a random number s between 1 and r.
- Proceed in the algorithm until you reached the F_s then stop.

This way we do not go back to a vertex we already visited.

What is the degree of w?

Again lets look at the randomization in a slightly different way:

- Order the facets we encounter by their maximal vertex value:

$$i > j \Rightarrow \max\{\phi(v) : v \in F_{(i)}\} \geq \max\{\phi(v) : v \in F_{(j)}\}$$

- In other words if we denote $v_{(i)}$ the optimal vertex of facet $F_{(i)}$ we have that $u(v_{(i)}) \leq n - i$.
- Starting from vertex v for which $u(v) \leq n$ with probability $\frac{1}{r}$ we reach a vertex w for which:

$$u(w) \leq n - i \quad \forall i = 1, \ldots, r$$

What is the degree of w in S_0?

- Notice that any vertex v has at most 1 inactive facet (only if the objective function is parallel to that facet)
- Assuming that all facets containing v are active with probabilities of $\frac{1}{d}$ we have:

$$u(w) \leq n - i \quad \forall i = 1, \ldots, d$$

- Assuming that v has one inactive facet then with probability $\frac{1}{d}$ $w = v$ and with probabilities of $\frac{1}{d}$ we have:

$$u(w) \leq n - i \quad \forall i = 1, \ldots, d - 1$$

- The latter case is of course worse.