WYKŁAD 20.05.2015

~~Pla~~ Analiza smoothed

Daniel Spielman, Shanghua Teng

Nagroda w 2008 roku, artykuł z 2004 roku (JACM), STOC 2001.

Nagroda jest za wprowadzenie nowego typu złożoności algorytmu, czegoś pomiędzy złożonością pesymistyczną, a złożonością średnią – złożoność smoothed oraz pokazanie, że złożoność smoothed algorytmu sympleks jest wielomianowa. Algorytm sympleks to praktycznie stosowany alg. rozwiązujący programowanie liniowe. Jego złożoność pesymistyczna jest wykładnicza, ale w praktyce działa bardzo dobrze. Ludzie od dawna zastanawiali się dlaczego tak jest, wydaje się, że analiza smoothed udzieliła właściwej odpowiedzi.

Zacznijmy wszystko od początku. Będziemy dużo mówić o algorytmie sympleks, a o analizie smoothed niewiele na końcu.

Problem programowania liniowego:

Zmaksymalizuj: $b^T x$

dla: $Ax \leq y$, $x \geq 0$.

Przykład (problem diety)

	cukry	białka	tłuszcze	żelazo	koszt
kromka chleba	30	5	1,5	10	30
Jogurt	10	9	2,5	0	80
Łyżka masła orzechowego	6	8	18	6	20
minimum wymagane	300	50	70	100	

Zminimalizuj:

$$30x_1 + 80x_2 + 20x_3$$

dla

$$30x_1 + 10x_2 + 6x_3 \geq 300$$

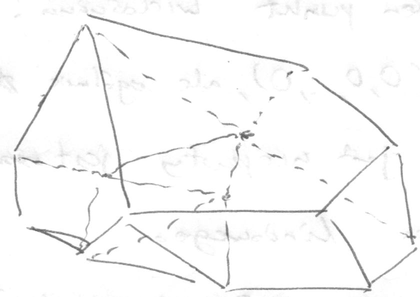
$$5x_1 + 9x_2 + 8x_3 \geq 50$$

$$1{,}5x_1 + 2{,}5x_2 + 18x_3 \geq 70$$

$$10x_1 + 0x_2 + 6x_3 \geq 100 \qquad x_1, x_2, x_3 \geq 0$$

Ogólnie to jest problem bardzo powszechny w praktyce. Tę instancję można przerobić na problem maksymalizacyjny.

W 1947 roku Dantzig zaproponował algorytm sympleks. On działa w następujący sposób. Obszar ograniczony nierównościami to pewien wielościan.



Łatwo pokazać, że jeśli maksymalizujemy funkcję liniową, to wówczas maksimum jest przyjmowane w pewnym wierzchołku.
Ponadto nasz wielościan jest wypukły. Jeśli w danym wierzchołku nie jest przyjmowane maksimum globalne, to ma jakiegoś sąsiada, który ma wyższą wartość funkcji celu. A zatem możemy postępować w następujący sposób. Zacznij w jakimś wierzchołku, jeśli któryś sąsiad ma wyższą funkcję celu, to idź tam, a jeśli nie, to już jestem w maksimum. Pozostaje jeszcze jedno pytanie do doprecyzowania algorytmu: jeśli wielu sąsiadów ma wyższą funkcję celu, to do którego idziemy?
Tu mogą być różne taktyki, one się nazywają po angielsku pivoting rules, po polsku nie wiem, powiedzmy, że reguła piwotowania. Dantzig zaproponował regułę: idź po krawędzi, która idzie najbardziej w kierunku celu.

Można jednak proponować inne reguły. Np. idź ~~...~~ do sąsiada, który ma ~~...~~ największą wartość ~~...~~ funkcji celu. Jest też sporo innych.

Tu pozostają pewne szczegóły techniczne. Jak znaleźć choćby jeden punkt wielościanu? W naszej sytuacji to jest proste $(0,0,\ldots,0)$, ale ogólnie ~~...~~ odpowiedź, czy wielościan jest niepusty jest ~~...~~ ~~z~~ równoważne rozwiązywaniu programowania liniowego.

Możemy bowiem połowić przedział: pokazać to!

Inne techniczne pytanie to: jak znaleźć wierzchołek sąsiada? To się chyba robi poprzez opuszczenie pewnej ściany i dodanie innej, ale tu też mogą być szczegóły techniczne.

Albo: Jak wielu jest sąsiadów.

Mamy więc pytanie: jak długo działa algorytm sympleks w pesymistycznym przypadku? Pierwsze pytanie brzmi:

Ile może być wierzchołków w wielościanie w $\mathbb{R}^d$ danym przez ~~...~~ n liniowych ograniczeń?

Pokazać, że może być ich wykładniczo wiele.

Przykładowo hiperkostka w ~~...~~ $\mathbb{R}^d$ ma $2d$ ograniczeń i 2^d wierzchołków. A jakie jest ograniczenie górne?

Każdy wierzchołek jest wyznaczony przez pewne d ograniczeń, czyli maksymalnie $\binom{n}{d}$.

Przez wiele lat ludzie próbowali pokazać, że sympleks działa w czasie wielomianowym aż wreszcie w 1972 roku Klee i Minty pokazali przykład, w którym działa wykładniczo długo.

Najpierw przeanalizujmy sympleks na sześcianie.
On robi co najwyżej 3 (ogólnie d) ruchy mimo, że wierzchołków jest 2^3 (2^d).

Okazuje się jednak, że jak się nieco zniekształci kostkę w d wymiarach, to już jest OK.

<u>Przykład Klee i Minty</u>

maksymalizuj $\displaystyle\sum_{j=1}^{d} 10^{d-j} x_j$

przy ograniczeniach: $\displaystyle 2\sum_{j=1}^{i-1} 10^{i-j} x_j + x_i \le 100^{i-1}$ dla $1 \le i \le d$

$x_i \geq 0$ dla $1 \le i \le d$

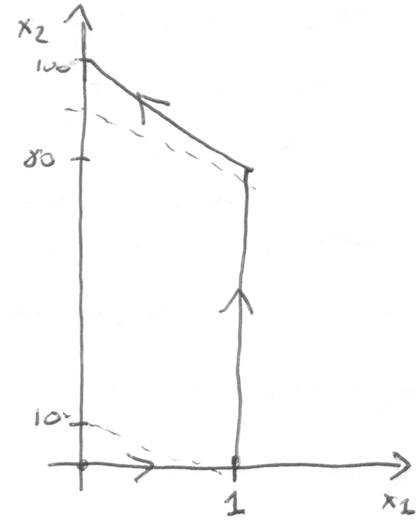
Przeanalizujmy to najpierw dla $d=2$.

Mamy:

maks. $10x_1 + x_2$

przy $x_1 \le 1$

$20x_1 + x_2 \le 100$



Teraz dla $d=3$

Mamy:

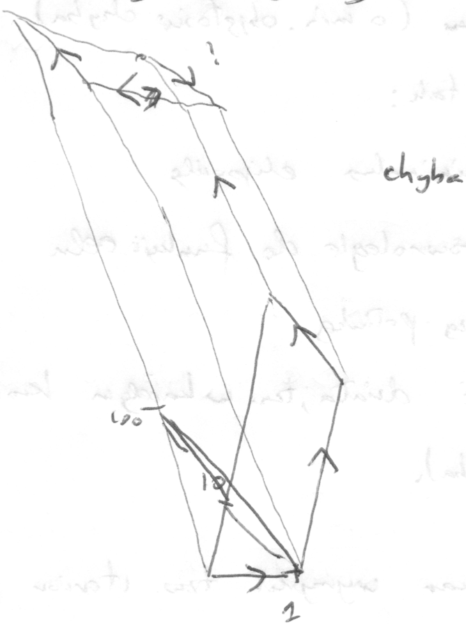
maks. $100x_1 + 10x_2 + x_3$

przy:

$$x_1 \le 1$$

$$20x_1 + x_2 \le 100$$

$$200x_1 + 20x_2 + x_3 \le 10000$$



Ogólnie on robi (chyba) $2^d - 1$ kroków.

Można zauważyć łatwo, że gdybyśmy zastosowali inną regułę piwotowania: idź do sąsiada o najwyższej funkcji celu, to poszlibyśmy od razu do maksimum.

Jednak dla wszystkich znanych reguł piwotowania opracowane zostały przykłady w stylu Klee i Minty'a. Do tej pory problemem otwartym jest, czy istnieje jakaś deterministyczna reguła piwotowania, przy której algorytm działa w czasie wielomianowym.

To nie znaczy, że nie jest znany wielomianowy algorytm dla programowania liniowego.

Są znane, przynajmniej dwa.

W 1979 roku Khachiyan wymyślił tzw. ellipsoid method. Bardzo z grubsza opiera się ona na tym, że istnieje wielomianowy algorytm znajdowania minimalnej elipsoidy ~~zaw~~ (o min. objętości chyba) zawierającej dany wielościan. Robimy tak:

- znajdujemy tą minimalną elipsoidę
- tniemy ją na pół równolegle do funkcji celu
- zajmujemy się tą lepszą połówką

Okazuje się, że to sensownie działa, tzn. w każdym kroku polepsza dokładność o coś (chyba).

W 1984 roku Karmarkar wymyślił tzw. interior point method. To polega bardzo z grubsza na tym, że zamiast po brzegu idziemy do środka wielościanu. Dodajemy pewną funkcję, która przy brzegu jest bardzo duża, a trochę dalej bardzo mała i odwracamy w kierunku gradientu (jak w metodzie Newtona). Robimy wiele iteracji.

To jest raczej moje wyobrażenie, może to działa nieco inaczej.

Tak, czy inaczej algorytm sympleks okazuje się najlepszy w praktyce. Ludzie długo myśleli dlaczego. Pokazano, że w przypadku średnim algorytm działa dobrze (oczekiwana złożoność), ale to nie było traktowane jako dobre wytłumaczenie. Przypadek średni to niekoniecznie przypadek typowy (patrz np. telewizor). Np. Borgwardt miał wynik

$O(m^3 d^{\frac{1}{n-1}})$.

Badano też zresztą losowe reguły piwotowania.

Pokazano coś w stylu ograniczenia $e^{C\sqrt{d \log d}}$.

Była też hipoteza Hirscha: maksymalna odległość pod. między wierzchołkami to $n-d$, ale została ona sfalsyfikowana niedawno. Przy czym nawet jej poprawność nie tłumaczyłaby wiele.

Teraz podejście smooth analysis. Idea to: zaburzamy nieco wejście i liczymy średnią.

$$C_{worst}(n) = \max_{x \in X_n} T(x)$$

$$C_{avg}(n) = \mathop{\mathbb{E}}_{x \in X_n} T(x)$$

$$C_{smooth}(n) = \max_{x \in X_n} \mathop{\mathbb{E}}_{\sigma \in \Gamma} T(x+\sigma)$$

Dla analizy smoothed na algorytmie sympleks użyto rozkładu gaussowskiego o średniej 0 i wariancji $\sigma \cdot \|x\|$, gdzie σ jest parametrem. Przypomnijmy gęstość tegoż.

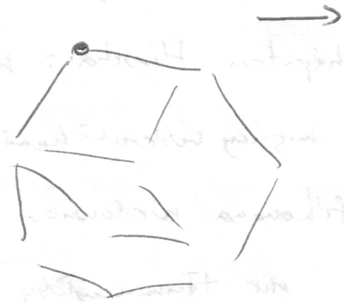
W $\mathbb{R}$ to jest

$$\frac{1}{\sqrt{2\pi} \cdot \sigma} \cdot e^{-\frac{x^2}{2\sigma^2}}$$

W $\mathbb{R}^d$ to jest $$\frac{1}{(\sqrt{2\pi} \cdot \sigma)^d} e^{-\frac{\|x\|^2}{2\sigma^2}}$$

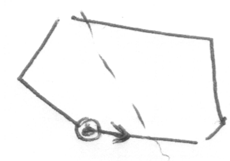
Oryginalna praca ma 94 strony, dużo rachunków itd.

W oryginalnej pracy zajęto się regułą pivotowania tzw. shadow vertex. Myśl jest taka:



Idę jakoś w kierunku [illegible]. W $\mathbb{R}^2$ jest tylko 1 taki kierunek dobry.

Tu biorę jakieś wejście i idę tam.
Okazuje się, że wierzchołki w cieniu to prawdziwe wierzchołki.



~~Każdy złożoności smooth~~

~~Alg~~

Pokazali, że złożoność (smoothed) alg. sympleks przy (chyba) tej regule pivotowania jest wielomianowa od $n, d, \frac{1}{\sigma}$.

Intuicja dlaczego to może być prawdziwe jest następująca.
Żeby ten algorytm trwał długo, to kąty powinny być małe, wtedy są mało zakrzywiony. Jeśli zrobimy losową perturbację, to te kąty nie będą wykładniczo małe.

Spielman i Teng udowodnili najpierw

Tw.

Niech $d \geq 3$, $n > d$, c i t są niezależnymi wektorami w $\mathbb{R}^d$, $a_1, \dots, a_n$ to gaussowskie wektory w $\mathbb{R}^d$ o wariancji $\sigma \leq \frac{1}{9 d \log n}$ o środkach w punktach o normie ≤ 1. Wtedy oczekiwana liczba wierzchołków w cieniu wielościanu $\{x : x^T a_i \leq 1\}$ na $\mathrm{Span}(t, c)$ to

$$\leq \frac{58\,888\,678\, n d^3}{\sigma^6}$$

To jeszcze nie daje wszystkiego, bo ~~[illegible]~~. tu t i c są ustalone, a potem losujemy a_i, podczas gdy ~~[illegible]~~ u nas jest odwrotnie.

Nie mniej jednak udało im się to doprowadzić do końca.

Zauważmy, że ~~[illegible]~~ analiza smooth rzeczywiście mówi dlaczego to jest praktyczne, dlaczego w praktyce działa dobrze. W praktyce dane są nieco zaburzone, pewnie o coś w stylu gaussowskiego.

Analiza smoothed była też potem stosowana do innych rzeczy jak: losowe chodzenie po losowym grafie, losowy problem plecakowy, losowe BST (wziąłem losowe problemy).