CSC2420: Algorithm Design, Analysis and Theory
Fall 2023
An introductory (i.e. foundational) level graduate course.

Allan Borodin

November 14, 2023

Week 9

Announcements:

- I have started to list questions for the third and final assignment. I will post two questions today. The assignment will be due Friday, December 8 at 11 AM.
- I have posted (on the web page) slides by Denis Pankratov on LP theory and LP duality.

Todays agenda

We will discuss the following topics:

- Finish up discussion of the random walk algorithm for 2SAT
- The random walk algorithm for kSAT
- Markov chains and random walks on a graph.
- Exponential Time Hypothesis (ETH) and Strong Exponential Time Hypothesis (SETH).
- The randomized algorithm for primality testing.
- The (weighted) vertex and set cover problems.
- Primal dual algorithms and primal dual fitting.

2SAT and kSAT using random walks

- We finish up our discuss of the randomized algorithm for 2-SAT (due to Papadimitriou [1991]) based on a random walk on the line graph with nodes $\{0, 1, \ldots, n\}$. We view being on node i as having a truth assignment τ that is Hamming distance i from some fixed satisfying assignment τ^* if such an assignment exists (i.e. F is satisfiable).
- Start with an arbitrary truth assignment τ and if $F(\tau)$ is true then we are done; else find an arbitrary unsatisfied clause C and randomly choose one of the two variables x_i occurring in C and now change τ to τ' by setting $\tau'(x_i) = 1 - \tau(x_i)$.

The expected time to reach a satisfying assignment

- When we randomly select one the the two literals in C and complement it, we are getting close to τ^* (i.e. moving one edge closer to node 0 on the line) with probability at least $\frac{1}{2}$. (If it turns out that both literal values disagree with τ^*, then we are getting closer to τ^* with probability $= 1$.)
- As we are proceeding in this random walk we might encounter another satisfying assignment which is all the better.
- It remains to bound the expected time to reach node 0 in a random walk on the line where on each random step, the distance to node 0 is reduced by 1 with probability at least $\frac{1}{2}$ and otherwise increased by 1 (but never exceeding distance n). This perhaps biased random walk is at least as good as the case where we randomly increase or decrease the distance by 1 with probability equal to $\frac{1}{2}$.

Claim:

The expected time to hit node 0 is at most $2n^2$.

- To prove the claim one needs some basic facts about Markov chains.

The basics of finite Markov chains

- A finite Markov chain M is a discrete-time random process defined over a set of states S and a matrix $P = \{P_{ij}\}$ of transition probabilities.
- Denote by X_t the state of the Markov chain at time t. It is a memoryless process in that the future behavior of a Markov chain depends only on its current state: $Prob[X_{t+1} = j | X_t = i] = P_{ij}$ and hence $Prob[X_{t+1} = j] = \sum_i Prob[X_{t+1} = j | X_t = i] Prob[X_t = i]$.
- Given an initial state i, denote by r_{ij}^t the probability that the first time the process reaches state j occurs at time t;
  $r_{ij}^t = Pr[X_t = j \text{ and } X_s \neq j \text{ for } 1 \leq s \leq t-1 | X_0 = i]$
- Let f_{ij} the probability that state j is reachable from initial state i;
  $f_{ij} = \sum_{t>0} r_{ij}^t$.
- Denote by h_{ij} the expected number of steps to reach state j starting from state i (hitting time); that is, $h_{ij} = \sum_{t>0} t \cdot r_{ij}^t$
- Finally, the *commute time* c_{ij} is the expected number of steps to reach state j starting from state i, and then return to i from j; $c_{ij} = h_{ij} + h_{ji}$

Stationary distributions

- Define $\mathbf{q}^t = (q_1^t, q_2^t, \ldots, q_n^t)$, the state probability vector (the distribution of the chain at time t), as the row vector whose i-th component is the probability that the Markov chain is in state i at time t.
- A distribution π is a stationary distribution for a Markov chain with transition matrix P if $\pi = \pi P$.
- Define the underlying directed graph of a Markov chain as follows: each vertex in the graph corresponds to a state of the Markov chain and there is a directed edge from vertex i to vertex j iff $P_{ij} > 0$. A Markov chain is *irreducible* if its underlying graph consists of a single strongly connected component. We end these preliminary concepts by the following theorem.

Theorem: Existence of a stationary distribution

For any finite, irreducible and aperiodic Markov chain,

(i) There exists a *unique* stationary distribution π.

(ii) For all states i, $h_{ii} < \infty$, and $h_{ii} = 1/\pi_i$.

Back to random walks on graphs

- Let $G = (V, E)$ be a connected, non-bipartite, undirected graph with $|V| = n$ and $|E| = m$. A uniform random walk induces a Markov chain M_G as follows: the states of M_G are the vertices of G; and for any $u, v \in V$, $P_{uv} = 1/deg(u)$ if $(u, v) \in E$, and $P_{uv} = 0$ otherwise.
- Denote by $(d_1, d_2, \ldots, d_n)$ the vertex degrees. M_G has a stationary distribution $(d_1/2m, \ldots, d_n/2m)$.
- Let $C_u(G)$ be the expected time to visit every vertex, starting from u and define $C(G) = \max_u C_u(G)$ to be the *cover time* of G.

Theorem: Aleliunas et al [1979]

Let G be a connected undirected graph. Then

1. For each edge (u, v), $C_{u,v} \leq 2m$,
2. $C(G) \leq 2m(n-1)$.

- Hence the 2-SAT random walk has expected time at most $2n^2$. to find a satisfying assignment in a satisfiable formula. Can use Markov inequality to obtain probability of not finding satisfying assignment.

Extending the random walk idea to k-SAT

- The random walk 2-Sat algorithm might be viewed as a drunken walk (and not an algorithmic paradigm). We could view the approach as a local search algorithm that doesn't know when it is making progress on any iteration but does have confidence that such an exploration of the local neighborhood is likely to be successful over time.
- We want to extend the 2-Sat algorithm to k-SAT. However, we know that k-SAT is NP-complete for $k \geq 3$ so our goal now is to improve upon the naive running time of 2^n, for formulas with n variables.
- In 1999, following some earlier results, Schöning gave a very simple (a good thing) random walk algorithm for k-Sat that provides a substantial improvement in the running time (over say the naive 2^n exhaustive search) and this is still almost the fastest (worst case) algorithm known.
- This algorithm was derandomized by Moser and Scheder [2011].
- Beyond the theoretical significance of the result, this is the basis for various Walk-Sat algorithms that are used in practice.

Schöning's k-SAT algorithm

The algorithm is similar to the 2-Sat algorithm with the difference being that one does not allow the random walk to go on too long before trying another random starting assignment. The result is a one-sided error alg running in time $\tilde{O}[(2(1 - /1k)]^n$; i.e. $\tilde{O}(\frac{4}{3})^n$ for 3-SAT, etc.

Randomized k-SAT algorithm

Choose a random assignment τ
Repeat 3n times % n = number of variables
If τ satisfies F then stop and accept
Else Else Let C be an arbitrary unsatisfied clause
  Randomly pick and flip one of the literals in C
End If

Claim

If F is satisfiable then the above succeeds with probability p at least $[(1/2)(k/k-1)]^n$. It follows that if we repeat the above process for t trials, then the probability that we fail to find a satisfying assignment is at most $(1-p)^t < e^{-pt}$. Setting $t = c/p$, we obtain error probability $(\frac{1}{e})^c$

Final comments on the complexity of k-SAT

- These random wals algorithm are the basis for WalkSat algorithms which employ all sorts of heuristics to obtain excellent results in practice for SAT.
- The random walk approach obtains a randomized 1-sided error algorithm (which in turn has a deterministic variant) for 3-SAT that runs in time $(1.324)^n$. This is, of course, still exponential but significantly better that 2^n.
- It is a big open question as to whether or not there is a $2^{o(n)}$ time algorithm for 3-SAT. **The exponential time hypothesis (ETH)**: There is no deterministic or randomized algorithm for 3-SAT running in time $2^{o(n)}$. This is an unproven conjecture even assuming $P \neq NP$.

Note: The notation time $\tilde{O}(c^n)$ ignores say n^k factors in the time bound.

Final comments on the complexity status of SAT

The random walk time bound bound $\tilde{O}[(2(1 - /1k)]^n$ can be stated as $O((c_k)^n)$ where $c_k \to 2$ as $k \to \infty$.

Strong exponential time hypothesis (SETH): There is no deterministic or randomized algorithm for SAT that runs in time c^n for any $c < 2$.

Perhaps surprisingly, the ETH and especially the SETH conjectures imply that for a number of polynomial time computable problems, rather simple algorithms provide approximately the best time bounds. This important observation led to a topic called *fine-grained complexity*.

For example, consider the following orthogonal vectors (OV) problem: Given a set S of n vectors over $\{0,1\}^d$ with $d = \omega(logn)$ (say $d = \lceil(\log_2 n)^2)\rceil$. Determine if S has a pair of orthogonal vectors.

R. Williams [2005] : SETH implies there is no 0-sided randomized algorithm for the OV problem having expected time $n^{2-\epsilon}$ for any $\epsilon > 0$. This in turn (using "fine-grained reductions") implies that the edit distance problem cannot be computed in time $n^{2-\epsilon}$.

The fine-grained landscape

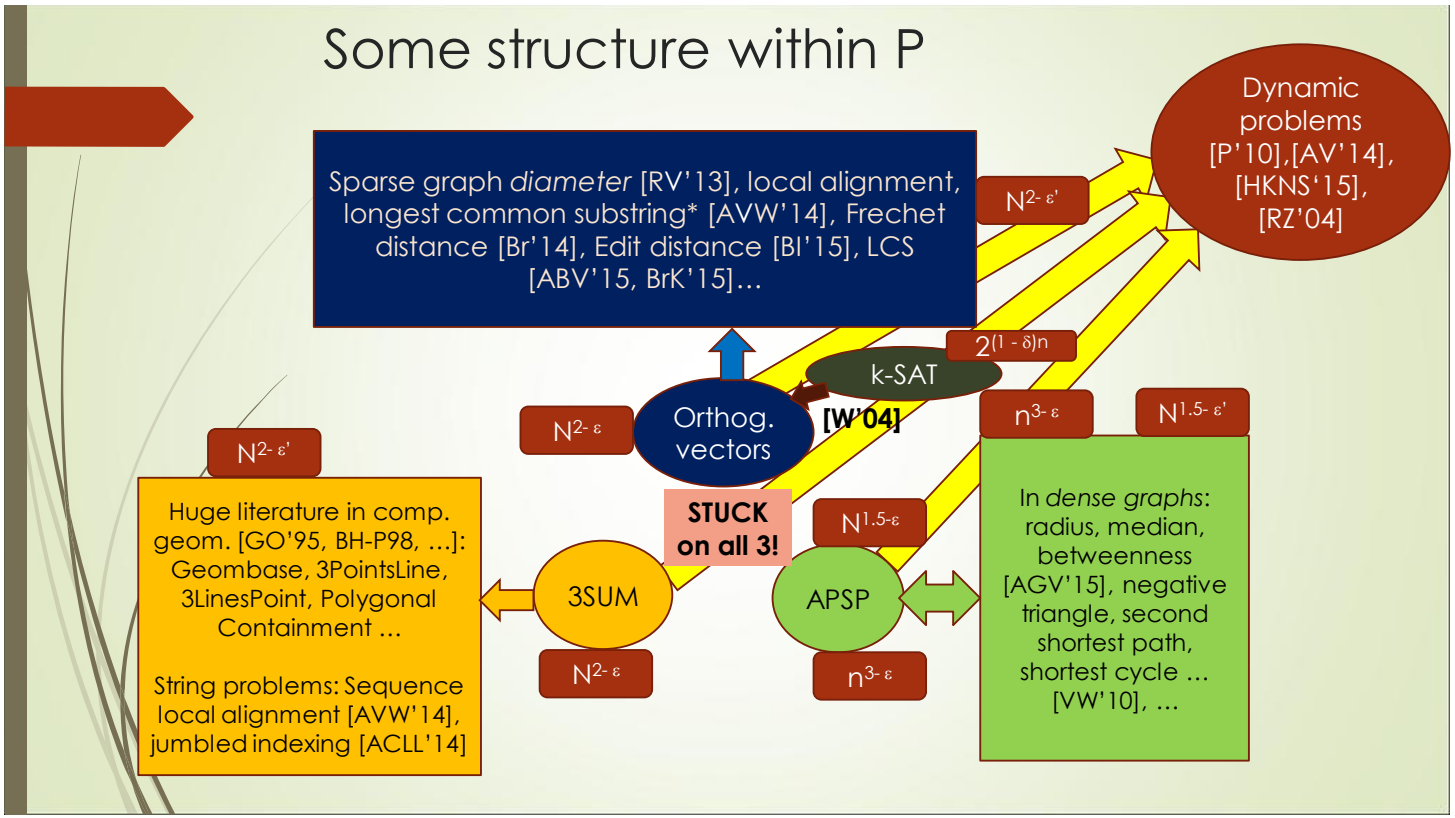

Figure: From V. Williams 2015

The all pairs shorest paths (APSP) problems

Even after (or because of) a number of relatively small improvements we still have the APSP conjecture.

APSP: given a weighted graph, find the **distance** between every two nodes.

Classical problem
Long history

APSP Conjecture: APSP on n nodes and O(log n) bit weights requires $n^{3-o(1)}$ time.

Author	Runtime	Year
Fredman	$n^3 \log\log^{1/3} n / \log^{1/3} n$	1976
Takaoka	$n^3 \log\log^{1/2} n / \log^{1/2} n$	1992
Dobosiewicz	$n^3 / \log^{1/2} n$	1992
Han	$n^3 \log\log^{5/7} n / \log^{5/7} n$	2004
Takaoka	$n^3 \log\log^2 n / \log n$	2004
Zwick	$n^3 \log\log^{1/2} n / \log n$	2004
Chan	$n^3 / \log n$	2005
Han	$n^3 \log\log^{5/4} n / \log^{5/4} n$	2006
Chan	$n^3 \log\log^3 n / \log^2 n$	2007
Han, Takaoka	$n^3 \log\log n / \log^2 n$	2012
Williams	$n^3 / \exp(\sqrt{\log n})$	2014

Primality testing

- I now want to briefly turn attention to one of the most influential randomized algorithms, namely a poly time randomized algorithm for primality (or perhaps better called compositeness) testing. Let $PRIME = \{N | N \text{ is a prime number}\}$ where N is represented in say binary (or any base other than unary) so that $n = |N| = O(\log N)$.
- History of polynomial time algorithms:
  1. Vaughan 1972 showed that *PRIMES* is in *NP*. Note that co-PRIMES (i.e. the composites) are easily seen to be in *NP*.
  2. One sided error randomized algorithms (for compositeness) by Solovay and Strassen and independently Rabin in 1974. That is, $Prob[ALG \text{ says } N \text{ prime } | N \text{ composite}] \leq \delta < 1$ and $Prob[ALG \text{ says } N \text{ composite } | N \text{ prime}] = 0$
  3. The Rabin test is related to an algorithm by Miller that gives a deterministic polynomial time algorithm assuming a conjecture that would follow from (the unproven) ERH. The Rabin test is now called the Miller-Rabin text.
  4. Goldwasser and Killian establish a 0-sided randomized algorithm.
  5. In 2002, Agrawal, Kayal and Saxena show that primality is in deterministic polynomial time.

Why consider randomized tests when there is a deterministic algorithm?

- Even though there is now a deterministic algorithm, it is not nearly as efficient as the 1-sided error algorithms which are used in practice. These randomized results spurred interest in the topic (and other number theoretic algorithms) and had a major role in cryptographic protocols (which often need random large primes). Moreover, these algorithms became the impetus for major developments in randomized algoritms.
- While many of our previous algorithms might be considered reasonably natural (or natural extensions of a deterministic algorithm), the primality tests require some understanding of the subject matter (i.e. a little number theory) and many number theoretical algorithms are not something that immediately comes to mind. You can judge the Miller-Rabin primality test if it is "reasonably natural".

Some basic number theory we need for primaltiy testing.

- $Z_N^* = \{a \in Z_N : gcd(a, N) = 1\}$ is a (commutative) group under multiplication mod N.
- If N is prime, then
  1. For $a \neq 0(modN), a^{N-1} = 1(modN)$.
  2. Z_N^* is a cyclic group; that is there exists a generator g such that $\{g, g^2, g^3, \ldots, g^{N-1}\}$ (all mod N) is the set Z_N^*. This implies that $g^i \neq 1(modN)$ for any $1 \leq i < N - 1$.
  3. There are exactly two square roots of 1 in Z_N^*, namely 1 and -1.
- The Chinese Remainder Theorem: Whenever N_1 and N_2 are relatively prime (i.e. $gcd(N_1, N_2) = 1$), then for all $v_1 < N_1$ and $v_2 < N_2$, there exists a unique $w < N_1 \cdot N_2$ such that $v_1 = w(modN_1)$ and $v_2 = w(modN_2)$.

A simple but "not quite" correct algorithm

We also need two basic computational facts.

1. a^i mod N can be computed efficiently.
2. $gcd(a, b)$ can be efficiently computed.

The following is a simple algorithm that works except for an annoying set of nunbers called Carmichael numbers.

Simple algorithm ignoring Carmichael numbers

Choose $a \in Z_N$ uniformly at random.
If $gcd(a, N) \neq 1$, then Output Composite
If a^{N-1} mod $N \neq 1$, then Output Composite
Else Output Prime

When does the simple algorithm work?

- $S = \{a \mid gcd(a, N) = 1 \text{ and } a^{N-1} = 1\}$ is a subgroup of Z_N^*
- If there exists an $a \in Z_N^*$ such that $gcd(a, N) = 1$ but $a^{N-1} \neq 1$, then S is a proper subgroup of Z_N^*.
- By Lagrange's theorem, if S is a proper subgroup, $|S|$ must divide the order of the group and thus $|S| \leq \frac{N-1}{2}$
- Thus the simple algorithm would be a 1-sided error algorithm with probabiltiy $< \frac{1}{2}$ of saying Prime when N is Composite.
- The only numbers that give us trouble are the Carmichael numbers (also known as *false primes*) for which $a^{N-1} = 1 mod N$ for all a such that $gcd(a, N) = 1$.
- It was only recently (relatively speaking) that in 1994 it was proven that there are an infinite number of Carmichael numbers.
- The first three Carmichael numbers are 561, 1105, 1729

Miller-Rabin 1-sided error algorithm

Let $N-1 = 2^t u$ with u odd %Since wlg. N is odd, $t \geq 1$
Randomly choose non zero $a \in Z_N$ %Hoping that a will be composite certificate
If $gcd(a, N) \neq 1$ then report Composite
$x_0 = a^u$ %All computation is done mod N
For $i = 1 \ldots t$
    $x_i := x_{i-1}^2$
    If $x_i = 1$ and $x^{i-1} \notin \{-1, 1\}$, then report Composite
End For
If $x_t \neq 1$, then report Composite $\%x^t = x^{N-1}$
Else report Prime

Analysis sketch of Miller-Rabin

- Let S be the set of $a \in N$ that pass (i.e. fool) the Rabin-Miller test. We want to show that S is a proper subgroup and then as before by Langrange we will be done.
- It suffices then to find one element $w \in Z_N^*$ that will not pass the Miller-Rabin test.
  Case 1: N is not Carmichael and then we are done.
  Case 2: N is Carmichael and hence N cannot be a prime power.
  - $N = N_1 \cdot N_2$ and $gcd(N_1, N_2) = 1$ and of course N odd
  - The non-certificates must include some b such that $b^{2^i u} = -1(modN)$ and hence $b^{2^i u} = -1(modN_1)$
  - By the Chinese Remainder Theorem, there exists $w = v(modN_1)$ and $w = 1(modN_2)$
  - Hence $w^{2^i u} = -1(modN_1)$ and $w^{2^i u} = 1(modN_2)$
  - This implies $w^{2^i u} \notin \{-1, 1\} \pmod N$

Weighted vertex cover: where the "natural greedy" is not that good

- We consider another example (weighted vertex cover) where the "natural greedy algorithm" does not yield a good approximation.
- The vertex cover problem: Given node weighted graph $G = (V, E)$, with node weights $w(v), v \in V$.
  Goal: Find a subset $V' \subset V$ that covers the edges (i.e. $\forall e = (u, v) \in E$, either u or v is in V') so as to mininize $\sum_{v \in V'} w(v)$.
- Even for unweighted graphs, the problem is known to be NP-hard to obtain a 1.3606 approximation and under another (not so universally believed) conjecture (UGC) one cannot obtain a $2 - \epsilon$ approximation.
- For the unweighted problem, there are simple 2-approximation greedy algorithms such as just taking V' to be any maximal matching.
- The set cover problem is as follows: Given a weighted collection of sets $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$ over an n element universe U with set weights $w(S_i)$.
  Goal: Find a subcollection $\mathcal{S}'$ that covers the universe so as to minimize $\sum_{S_i \in \mathcal{S}'} w(S_i)$.

The natural greedy algorithm for weighted set cover

"The natural" greedy algorithm for weighted set cover

$\mathcal{S}' = \varnothing$

While there are uncovered elements in the universe U

Let $j = argmin_i\{w(S_i)/|S_i \cap U|$

$\mathcal{S}' = \mathcal{S}' \cup \{S_i\}$

$U = U \setminus \{S_i\}$

End While

- The vertex and set cover problems were two of Karp's NP-complete problems.
- Johnson[1974] and Lovasz[1975] independently showed that this natural greedy algorithm provides a $H(m) \approx \ln m$ approximation for the unweighted case where $m = \max_i |S_i| \leq |U|$. This was extended by Chvatal[1979] to the weighted case.
- Under a reasonable complexity assumption, Feige[1979] showed that it was not possile to acheive a $(1 - \epsilon) \ln n$ approximation even for the unweighted case.

The natural greedy algorithm for weighted vertex cover (WVC)

Vertex cover can be viewed as a special case of set cover. (How?)

The natural greedy algorithm for weighted vertex cover (WVC)

Vertex cover can be viewed as a special case of set cover. (How?) Then the natural greedy set cover algorithm (which is essentially optimal for set cover up to standard complexity assumptions) becomes the following:

$d'(v) := d(v)$ for all $v \in V$

% $d'(v)$ will be the residual degree of a node

While there are uncovered edges

Let v be the node minimizing $w(v)/d'(v)$

Add v to the vertex cover;

remove all edges in $Nbhd(v)$;

recalculate the residual degree of all nodes in $Nbhd)v)$

End While

Figure: Natural greedy algorithm for weighted vertex cover. Approximation ratio $H_n \approx \ln n$ where $n = |V|$.

Clarkson's [1983] modified greedy for WVC

$d'(v) := d(v)$ for all $v \in V$

% $d'(v)$ will be the residual degree of a node

$w'(v) := w(v)$ for all $v \in V$

% $w'(v)$ will be the residual weight of a node

While there are uncovered edges

Let v be the node minimizing $w'(v)/d'(v)$

$w := w'(v)/d'(v)$

$w'(u) := w'(u) - w$ for all $u \in Nbhd(v)$

% For analysis only, set $we(u, v) = w$

Add v to the vertex cover;

remove all edges in $Nbhd(v)$;

recalculate the residual degree of all nodes in $Nbhd(v)$

End While

Figure: Clarkson's greedy algorithm for weighted vertex cover. Approximation ratio 2.

Solving the f-frequency set cover

In the f-frequency set cover problem, each element is contained in at most f sets. We can solve the f-frequency cover problem by obtaining an optimal solution $\{x_j^*\}$ to the (primal) LP and then rounding to obtain $\bar{x}_j = 1$ iff $x_j^* \geq \frac{1}{f}$.

The set cover problem as an IP/LP

minimize $\sum_j w_j x_j$
subject to $\sum_{j:u_i \in S_j} x_j \geq 1$ for all i; that is, for all $u_i \in U$
$x_j \in \{0,1\}$ (resp. $x_j \geq 0$)

This is a conceptually simple method but requires solving the LP. We will see that the primal dual method allows us to acheieve the same approximation without solving the LP.

End of November 14 class

We ended here but I am posting remaining slides as they may help for the assignment.

Duality: See Vazirani and Shmoys/Williamson texts, and Williamson article

- For a primal maximization (resp. minimization) LP in standard form, the dual LP is a minimization (resp. maximization) LP in standard form.
- Specifically, if the primal $\mathcal{P}$ is:
  - Minimize $\mathbf{c} \cdot \mathbf{x}$
  - subject to $A_{m \times n} \cdot \mathbf{x} \geq \mathbf{b}$
  - $\mathbf{x} \geq 0$
- then the dual LP $\mathcal{D}$ with dual variables $\mathbf{y}$ is:
  - Maximize $\mathbf{b} \cdot \mathbf{y}$
  - subject to $A^{tr}_{n \times m} \cdot \mathbf{y} \leq \mathbf{c}$
  - $\mathbf{y} \geq 0$
- Note that the dual (resp. primal) variables are in correspondence to primal (resp. dual) constraints.
- If we consider the dual $\mathcal{D}$ as the primal then its dual is the original primal $\mathcal{P}$. That is, the dual of the dual is the primal.

An example: set cover

The set cover problem as an IP/LP

minimize $\sum_j w_j x_j$
subject to $\sum_{j:u_i \in S_j} x_j \geq 1 \quad$ for all $u_i \in U$
$x_j \in \{0,1\}$ (resp. $x_j \geq 0$)

The dual LP

maximize $\sum_i y_i$
subject to $\sum_{i:u_i \in S_j} y_i \leq w_j \quad$ for all j
$y_i \geq 0$

If all the parameters in a standard form minimization (resp. maximization) problem are non negative, then the problem is called a covering (resp. packing) problem. Note that the set cover problem is a covering problem and its dual is a packing problem.

Duality Theory Overview

- An essential aspect of duality is that a finite optimal value to either the primal or the dual determines an optimal value to both.
- The relation between these two can sometimes be easy to interpret. However, the interpretation of the dual may not always be intuitively meaningful.
- Still, duality is very useful because the duality principle states that optimization problems may be viewed from either of two perspectives and this might be useful as the solution of the dual might be much easier to calculate than the solution of the primal.
- In some cases, the dual might provide additional insight as to how to round the LP solution to an integral solution.
- Moreover, the relation between the primal $\mathcal{P}$ and the dual $\mathcal{D}$ will lead to primal-Dual algorithms and to the so-called dual fiiting analysis.
- In what follows we will assume the primal is a minimization problem to simplify the exposition.

Strong and Weak Duality

Strong Duality

If x^* and y^* are (finite) optimal primal and resp. dual solutions, then $\mathcal{D}(\mathbf{y}^*) = \mathcal{P}(\mathbf{x}^*)$.

Note: Before it was known that solving LPs was in polynomial time, it was observed that strong duality proves that LP (as a decision problem) is in $\mathbf{NP} \cap \mathbf{co} - \mathbf{NP}$ which strongly suggested that LP was not NP-complete.

Weak Duality for a Minimization Problem

If $\mathbf{x}$ and $\mathbf{y}$ are primal and resp. dual solutions, then $\mathcal{D}(\mathbf{y}) \leq \mathcal{P}(\mathbf{x})$.

- Duality can be motivated by asking how one can verify that the minimum in the primal is at least some value z.

Motivating duality

Consider the motivating example in V. Vazirani's text:

Primal

minimize $7x_1 + x_2 + 5x_3$

subject to

- (1) $x_1 - x_2 + 3x_3 \geq 10$
- (2) $5x_1 + 2x_2 - x_3 \geq 6$
- $x_1, x_2, x_3 \geq 0$

Dual

maximize $10y_1 + 6y_2$

subject to

$y_1 + 5y_2 \leq 7$

$-y_1 + 2y_2 \leq 1$

$3y_1 - y_2 \leq 5$

$y_1, y_2 \geq 0$

Adding (1) and (2) and comparing the coefficient for each x_i, we have:

$7x_1 + x_2 + 5x_3 \geq (x_1 - x_2 + 3x_3) + (5x_1 + 2x_2 - x_3) \geq 10 + 6 = 16$

Better yet,

$7x_1 + x_2 + 5x_3 \geq 2(x_1 - x_2 + 3x_3) + (5x_1 + 2x_2 - x_3) \geq 26$

For an upper bound, setting $(x_1, x_2, x_3) = (7/4, 0, 11/4)$

$7x_1 + x_2 + 5x_3 = 7 \cdot (7/4) + 1 \cdot 0 + 5 \cdot (11/4) = 26$

This proves that the optimal value for the primal and dual solution $(y_1, y_2) = (2, 1)$ must be 26.

Easy to prove weak duality

The proof for weak duality

$$\begin{aligned}
&\mathbf{b} \cdot \mathbf{y} = \sum_{j=1}^{m} b_j y_j \\
&\leq \sum_{j=1}^{m} (\sum_{i=1}^{n} A_{ji} x_i) y_j \\
&\leq \sum_{i=1}^{n} \sum_{j=1}^{m} (A_{ji} y_j) x_i \\
&\leq \sum_{i=1}^{n} c_i x_i = \mathbf{c} \cdot \mathbf{x}
\end{aligned}$$

Primal dual for f-frequency set cover

We know that for a minimization problem, any dual solution is a lower bound on any primal solution. One possible goal in a primal dual method for a minimization problem will be to maintain a fractional feasible dual solution and continue to try improve the dual solution. As dual constraints become tight we then set the corresponding primal variables.

Suggestive lemma

Claim: Let $\{y_i^*\}$ be an optimal solution to the dual LP and let $\mathcal{C}' = \{S_j | \sum_{e_i \in S_j} y_i^* = w_j\}$. Then $\mathcal{C}'$ is a cover.

Primal dual for f-frequency set cover continued

This suggests the following algorithm:

Primal dual algorithm for set cover

Set $y_i = 0$ for all i; $\mathcal{C}' := \varnothing$
While there exists an e_i not covered by $\mathcal{C}'$
    Increase the dual variables y_i until there is some j : $\sum_{\{k:e_i \in S_j\}} y_i = w_j$
    $\mathcal{C}' := \mathcal{C}' \cup \{S_j\}$
    Freeze the y_i associated with the newly covered e_i
End While

Theorem: Approximation bound for primal dual algorithm

The cover formed by tight constraints in the dual solution provides an f approximation for the f-frequency set cover problem.

Comments on the primal dual algorithm

- What is being shown is that the integral primal solution is within a factor of f of the dual solution which implies that the primal dual algorithm is an f-approximation algorithm for the f-frequency set cover problem.
- In fact, what is being shown is that the integrality gap of this IP/LP formulation for f-frequency set cover problem is at most f.
- In terms of implementation we would calculate the minimum ϵ needed to make some constraint tight so as to chose which primal variable to set. This ϵ could be 0 if a previous iteration had more than one constraint that becomes tight simultaneously. This ϵ would then be subtracted from w_j for j such that $e_i \in S_j$.

More comments on primal dual algorithms

- We have just seen an example of a basic form of the primal dual method for a minimization problem. Namely, we start with an infeasible integral primal solution and feasible (fractional) dual. (For a covering primal problem and dual packing problem, the initial dual solution can be the all zero solution.) Unsatisfied primal constraints suggest which dual constraints might be tightened and when one or more dual constraints become tight this determines which primal variable(s) to set.
- Some primal dual minimization algorithms extend this basic form by using a second (reverse delete) stage to achieve minimality. Some primal dual maximization algorithms use a reverse delete to enforce feasibility. There is some (for me not precise) relation between primal dual and local ratio algorithms (see Bar-Yehuda and Rawitz)
- **NOTE:** In the primal dual method we are not solving any LPs. Primal dual algorithms are viewed as "combinatorial algorithms" and in some cases they might even suggest an explicit greedy algorithm.

Using dual fitting to prove the approximation ratio of the greedy set cover algorithm

We have already seen the following natural greedy algorithm for the weighted set cover problem:

The greedy set cover algorithm

$\mathcal{C}' := \varnothing$

While there are uncovered elements

Choose S_j such that $\frac{w_j}{|\tilde{S}_j|}$ is a minimum where

$\tilde{S}_j$ is the subset of S_j containing the currently uncovered elements

$\mathcal{C}' := \mathcal{C}' \cup S_j$

End While

We wish to prove the following theorem (Lovasz[1975], Chvatal [1979]):

Approximation ratio for greedy set cover

The approximation algorithm for the greedy algorithm is H_d where d is the maximum size of any set S_j.

The dual fitting analysis

The greedy set cover algorithm setting prices for each element

$\mathcal{C}' := \varnothing$

While there are uncovered elements

Choose S_j such that $\frac{w_j}{|\tilde{S}_j|}$ is a minimum where

$\tilde{S}_j$ is the subset of S_j containing the currently uncovered elements

%Charge each element e in $\tilde{S}_j$ the average cost $price(e) = \frac{w_j}{|\tilde{S}_j|}$

% This charging is just for the purpose of analysis

$\mathcal{C}' := \mathcal{C}' \cup S_j$

End While

- We can account for the cost of the solution by the costs imposed on the elements; namely, $\{price(e)\}$. That is, the cost of the greedy solution is $\sum_e price(e)$.

Dual fitting analysis continued

- The goal of the dual fitting analysis is to show that $y_e = price(e)/H_d$ is a feasible dual and hence any primal solution must have cost at least $\sum_e price(e)/H_d$.
- Consider any set $S = S_j$ in $\mathcal{C}$ having say $k \leq d$ elements. Let $e_1, \ldots, e_k$ be the elements of S in the order covered by the greedy algorithm (breaking ties arbitrarily). Consider the iteration is which e_i is first covered. At this iteration $\tilde{S}$ must have at least $k - i + 1$ uncovered elements and hence S could cover cover e_i at the average cost of $\frac{w_j}{k-i+1}$. Since the greedy algorithm chooses the most cost efficient set, $price(e_i) \leq \frac{w_j}{k-i+1}$.
- Summing over all elements in S_j, we have $\sum_{e_i \in S_j} y_{e_i} = \sum_{e_i \in S_j} price(e_i)/H_d \leq \sum_{e_i \in S_j} \frac{w_j}{k-i+1} \frac{1}{H_d} = w_j \frac{H_k}{H_d} \leq w_j$. Hence $\{y_e\}$ is a feasible dual.