CSC2420: Algorithm Design, Analysis and Theory Spring (or Winter for pessimists) 2017

Allan Borodin

March 6, 2017

Lecture 8

Announcements:

- Assignment 1 is graded and I will bring graded assignments to class.
- I have posted three initial questions for Assignment 2.
- I have been reserving Thursdays 3-4 for an office hour but since there is little interest in the office hour, it seems best to simply email me and let me know when you would like to see me. I welcome discussing the course whenever I am free.

Todays agenda:

- Some concluding remarks on lecture L7 given by Lalla Mouatadid
- Continue randomized algorithms.
  1. Discussion of online bipartite matching and the KVV algoritihm.
  2. Some comments on extensions and variations of the online bipartite matching problem.
  3. Further comments regarding deterministic and randomized algorithms.
  4. Random walks

Some concluding comments on L7

Lecture 7 (given by Lalla Mouatadid) concerned two related topics, namely what is and is not FTP (a fixed tractable problem), and the topic of fine-grained complexity.

Both topics introduce more nuanced reductions so as to better understand the complexity of many search and optimization problems. In both topics, the work to date has mainly concerned optimal algorithms but one can also take the same perspective for approximation algorithms.

The focus for the FTP topic is usually NP-hard problems and the goal is to try to identify parameters that (for some problems) make the problem tractable (i.e. computable in polynomial time) when one fixes the parameter. That is, when the parameter value is k and n is the size of the problem, one can obtain an optimal algorithm in time $f(k)poly(n)$. The FTP toipic is by now a well established topic with both positive and negative results (i.e. assumptions that show when a problem is not FTP or show bounds on how fast f must grow).

Concluding comments on L7 continued

Although examples have existed for some long time, fine-grained complexity is a relatively new general topic of interest within TCS. The work to date has been mainly to provide reductions giving evidence that the complexity of known (and usually relatively simple) polynomial time algorithms cannot be improved.

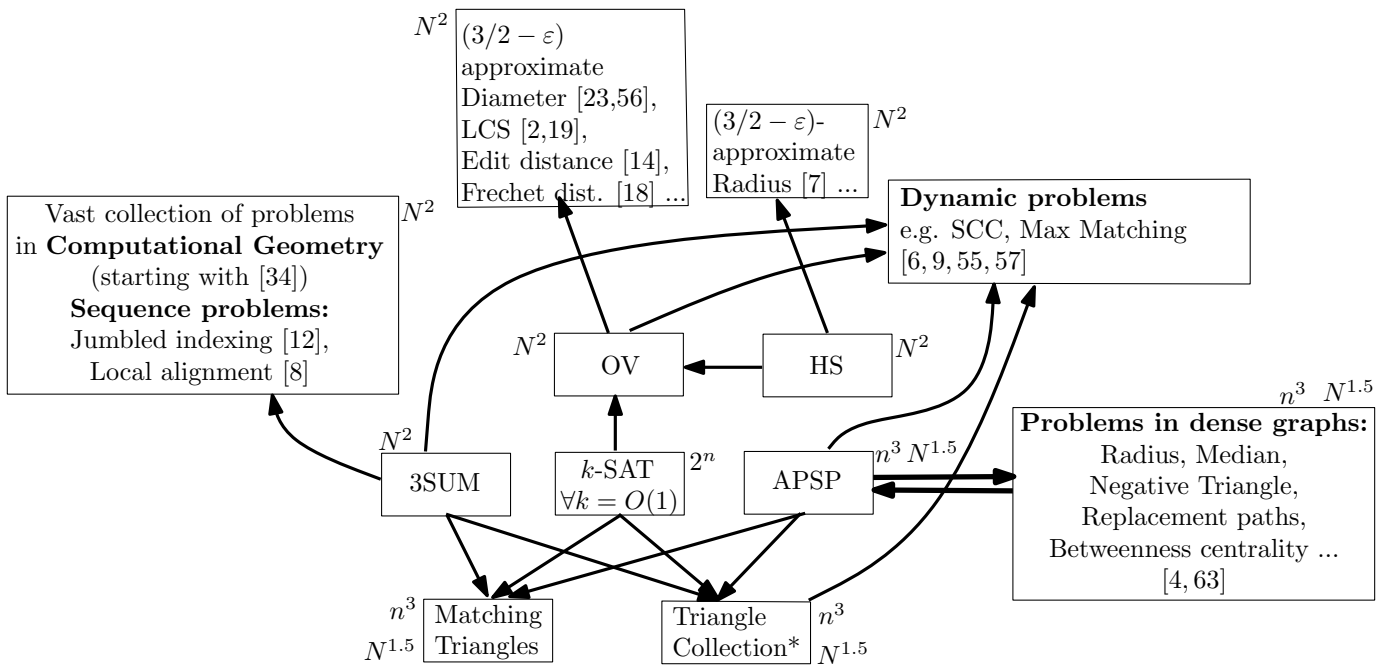

Figure : Known fine-grained reductions from Virginia V. Williams [2015] survey

Revisiting the KVV randomized online bipartite matching algorithm

We recall the online unweighted bipartite matching problem.

- We have a bipartite graph G with nodes $U \cup V$; nodes in U enter online revealing all their edges. A deterministic greedy matching produces a maximal matching and hence a $\frac{1}{2}$ approximation.
- We again claim that it is easy to see that any deterministic online algorithm cannot be better than a $\frac{1}{2}$ approximation even when the degree of every $u \in U$ is at most 2.
- The Karp, Vazirani and Vazirani (KVV) algorithm randomily orders the offline nodes in V and then when an online node $u \in U$ appears, it matches u to the highest ranked unmatched $v \in V$ such that (u, v) is an edge (if such a v exists).
- Equivalently, this algorithm can be viewed as a deterministic greedy (i.e. always matching when possible and breaking ties consistently) algorithm in the ROM model.

The worst case adversarial inapproximation

We will consider two proofs of the KVV approximation but first it is worth noting that no deterministic or randomized algorithm can (asymptotically) achieve an approximation better than $1 - 1/e$.

The worst case adversarial inapproximation

We will consider two proofs of the KVV approximation but first it is worth noting that no deterministic or randomized algorithm can (asymptotically) achieve an approximation better than $1 - 1/e$.

To derive negative results (e.g. inapproximations or lower bounds on time complexity) for randomized algorithms for online or offline algorithms, one often appeals to the *Yao Principle*.

Yao Principle for online algorithm approximations

Let $\mathcal{A}$ be a class of randomized online algorithms. To obtain a lower bound on the worst case approximation ratio for inputs with n input items, it is sufficient to find a distribution on input sequences and derive a bound for all deterministic algorithms iin the class $\mathcal{A}$. The Yao Principle is a consequence of the von Neumann minimax theorem for zero sum games.

For example, consider the online (n, n) biparitite problem. Consider the graph represented by an upper triangular matrix. It can be shown that as $n \to \infty$, the expected (wrt. column permutations) approximation ratio for any deterministic online algorithm limits (from above) to $1 - 1/e$.

Alternative proofs of the KVV result

As discovered by Goel and Mehta [08] (and independently discovered but unpublished by Krohn and Varadarajan), there was a flaw (in one of the main lemmas) in the KVV analysis of the Ranking algorithm. Goel and Mehta and ohers have now provided proofs of the KVV result.

- Goel and Mehta [08] (in the context of a more general problem) gave the first correct proof using a "factor revealing LP".
- Birnbaum and Mathieu [08] gave a simplified purely combinatorial proof.
- Devanur, Jain and Kleinberg gave what they believed to be "the simplest analysis yet" using a dual fitting analysis. The hope is that this analysis might lead to an analysis of the online budgetted allocation problem.
  Note: There is something both elegant but yet (without understanding previous work) mysterious in the Devanur et al analysis of the KVV algorithm.

Comments on the Birnbaum and Mathieu proof

- The worst case examples are (n, n) graphs with a perfect matching.
- In particular, for $n = 2$, the precise expected approximation ratio is $\frac{3}{4}$. The inapproximation can be seen by using the Yao principle and the approximation is a simple analysis.

The main lemma in the analysis

Let x_t be the probability (over the random permutations of the vertices in V) that the vertex of rank t is matched. Then the probability that the vertex of rank t is not matched is $1 - x_t \leq \frac{1}{n} \sum_{s=1}^{t} x_s$

- Letting $S_t = \sum_{s=1}^{t} x_s$ the lemma can be restated as $S_t(1 + 1/n) \geq 1 + S_{t-1}$ for all t. Given that the graph has a perfect matching, the expected competitive ratio is the infimum S_n/n. We set all inequalities to equalities and solve the recurrence. It is shown that $\frac{1}{n} S_n \geq 1 - (1 - \frac{1}{n+1})^n \to 1 - 1/e$ from below (whereas the known inapproximation approaches $1 - 1/e$ from above.

The proof of the main lemma in the Birnbaum and Mathieu proof

The proof of the main lemma depends on a "technical modification" of the following simple lemma:

Simple lemma about KVV Ranking algorithm

Let the mapping $m^* : U \to V$ be a perfect matching and let σ be a (random) permutation of the offline vertices V. Let $m^*(u) = v$. Then if KVV does not match v, it must match u to some v' with rank $\sigma(v') < \sigma(v)$.

Let R_{t-1} be the vertices of U of rank at most $t-1$ that are matched by KVV. **If** u and R_{t-1} were independent then the main lemma would be slightly better as it would immediately follow that $1 - x_t \leq \sum_{1 \leq s \leq t-1} x_s$ since the event $u \in R_{t-1}$ would have probability $\frac{R_{t-1}}{n}$.

The proof of the main lemma in the Birnbaum and Mathieu proof

The proof of the main lemma depends on a "technical modification" of the following simple lemma:

Simple lemma about KVV Ranking algorithm

Let the mapping $m^* : U \to V$ be a perfect matching and let σ be a (random) permutation of the offline vertices V. Let $m^*(u) = v$. Then if KVV does not match v, it must match u to some v' with rank $\sigma(v') < \sigma(v)$.

Let R_{t-1} be the vertices of U of rank at most $t-1$ that are matched by KVV. **If** u and R_{t-1} were independent then the main lemma would be slightly better as it would immediately follow that $1 - x_t \leq \sum_{1 \leq s \leq t-1} x_s$ since the event $u \in R_{t-1}$ would have probability $\frac{R_{t-1}}{n}$.

But u is not indpendent of R_{t-1} so that is the need for the technical modification that will render u independent of the offline permutation σ that ranks the offline vertices.

The Devanur et al primal dual analsyis of KVV

The offline vertices in L (resp. online vertices in R) will be referred to by indices i (resp. j). The KVV algorithm can be restated as follows:

Algorithm 1: The RANKING algorithm.

foreach $i \in L$ **do**
  Pick $Y_i \in [0, 1]$ uniformly at random
foreach $j \in R$ **do**
  When j arrives, let $N(j)$ denote the set of unmatched neighbors of j;
  if $N(j) = \emptyset$ **then**
    j remains unmatched
  else
    Match j to $\arg\min\{Y_i : i \in N(j)\}$

The LP relaxation and the dual of the standard IP for bipartite matching are :

maximize $\sum_{(i,j)\in E} x_{ij}$ minimize $\sum_{i\in L} \alpha_i + \sum_{j\in R} \beta_j$ subject to

$\sum_{j:(i,j)\in E} x_{ij} \leq 1 \; \forall \in L$ $\alpha_i + \beta_j \geq 1 \; \forall (i,j) \in E$

All variables are non negative.

Devanur et al KVV analysis continued

The dual variables α_i β_j will be set randomly (as a function of the random Y_i in the algorithm) and the primal x_{ij} variables will be set integrally to insure a feasible matching.

The essential features of the analysis are:

- For all unmatched i and j by KVV, $x_{ij} = \alpha_i = \beta_j = 0$
- If j is matched to i by KVV, then $x_{ij} = 1, \alpha_i = g(Y_i)/F$ and $\beta_j = 1 - \alpha_i$ where (the somewhat mysterious) $g(y) = e^{y-1}$ and $F = 1 - \frac{1}{e}$ is the desired approximation ratio.
- Setting the primal to $x_{ij} = 1$ then corresponds to setting the corresponding dual $\alpha_i + \beta_j = 1/F$. This insures that the value of the dual "solution" is equal to $1/F \cdot$ (value of the constructed match).
- If the dual solution were feasible then by the usual primal dual argument, the approximation would be an F approximation since $IP - OPT \leq LP - OPT = DUAL - OPT \leq$ computed dual $= \frac{1}{F} \cdot$ KVV solution

Completing the sketch of the Devanur et al analysis

- However, the dual "solution" is not necessarily feasible. But what can be shown is that the expectation (over the $\{Y_i\}$) is feasible for every $(i,j) \in E$ which implies that the desired result that KVV is an F approximation in expectation.
- The technical part of the analysis is to show :

  LEMMA 2.4. *If g and F are such that*

  $$(2.1) \qquad \forall\ \theta \in [0,1] \int_0^\theta g(y)\,dy + 1 - g(\theta) \geq F,$$

  then the duals constructed are feasible in expectation.

- The proof is completed by showing that for all $(i,j) \in E$
  1. $\mathbb{E}_{Y_i}[\alpha_i] \geq \int_0^\theta g(y)dy/F$ and
  2. $\mathbb{E}_{Y_i}[\beta_j] \geq (1 - g(\theta))/F$
  3. Hence $\mathbb{E}_{Y_i}[\alpha_i + \beta_i] \geq 1$

Getting past the $(1 - 1/e)$ bound: stochastic optimization

- The ROM model is an example of stochastic optimization. There are other stochastic optimization models that are perhaps more naturally studied, namely sampling inputs from known and unknown distributions. In particular, for online algorithms we can consider the i.i.d model where each online input item is independently drawn from some (known or unknown) distribution.
- The KVV algorithm can be interpreted as a deterministic algorithm (which we can call *Fixed Ranking*) in the ROM model and hence achieveing the $1 - 1/e$ approximation. For the ROM model, it is not known if there ia a deterministic online algorithm with approximation better than $1 - 1/e$.
- Karande et al [2011] show that the randomized KVV Ranking algorithm achieves approximation .653 in the ROM model and is no better than .727.

Ranking in the ROM model continued

- Mahdian and Yan improve the Karande et al bound to achieve approximation .696 for Ranking in the ROM model. Their analysis uses a family of "strongly factor revealing LPs".
- Karande et al also show that when the adversarial graph has many perfect (or near perfect) matchings, the approximation is much improved. They observe that this result explains "a puzzling mystery" as to simulations of Ranking on the nemesis graph for Ranking in the adversarial input model (i.e. the graph represented by an upper triangular adjacency matrix) showing that Ranking has ratio no better than $1 - 1/e$. This graph has $\omega(1)$ almost perfect matchings and hence Ranking will have an expected ratio approaching 1 in the ROM model on this nemesis graph.
- Karande, et al [2011] also show that an approximation in the ROM model implies the same approximation in the unknown .i.d. distribution model.
- These results for the ROM model follow earlier work on the known i.i.d. model starting with Feldman et al [2009].

ROM approximation implies unknown i.i.d. approximation

The Kanada observation is short and simple but not at all obvious until it was stated. And this useful observation applies to any online problem.

Suppose we have an unknown i.i.d. distribution D. Partition the possible input sequences into classes, where each class contains the same multi set of input items. Furthermore, each class contains $n!$ sequences each occuring with equal probability (since D was i.i.d.). Given that we have (say) an algorithm $\mathcal{A}$ with an α approximation in the ROM model, then $\mathcal{A}$ obtains an α approximation for each class and then we obtain the desired result by taking the expectation over all classes.

ROM approximation implies unknown i.i.d. approximation

The Kanada observation is short and simple but not at all obvious until it was stated. And this useful observation applies to any online problem.

Suppose we have an unknown i.i.d. distribution D. Partition the possible input sequences into classes, where each class contains the same multi set of input items. Furthermore, each class contains $n!$ sequences each occuring with equal probability (since D was i.i.d.). Given that we have (say) an algorithm $\mathcal{A}$ with an α approximation in the ROM model, then $\mathcal{A}$ obtains an α approximation for each class and then we obtain the desired result by taking the expectation over all classes.

It follows that the randomized Ranking algorithm obtains a .696 approximation in the unknown and known i.i.d. models. Unlike many of the i.i.d. approximations, the current ROM approximations do not require knowing n nor make any further assumptions (other than i.i.d.) about the distribution. As far as I know, there are no other approximation results for online bipartite matching in the i.i.d unknown distribution model.

Results for the (known) i.i.d model

- Beyond the usual issues that for online algorithms (e.g. is the number n of online input items known), there are some issues particular to the i.i.d models. Namely, do results depend on the size of the support of the distribution, and does the distribution have "integer type"?
- Feldman et al [2009] study the known distribution case and show a randomized algorithm that first computes an optimal offline solution (in terms of expectation) and uses that to guide an online allocation.
- They achieve a .67 approximation (improved to .699 by Bahmani and Kapralov [2010] and also show that no online algorithm can achieve better than 26/27 (improved by Bahmani and Kapralov to .902).
- Further improvements in the known i.i.d. approximations are:
  1. Manshadi et al [2011]]: .706 for integer (resp. .702 for arbitrary type) and polynomial support distributions.
  2. Jaillet and Lu [2012]: .725 for integer type (resp. .706 for arbitarry type). They also introuce a Poisson i.i.d. arrival process.
  3. Bruback et al [2016: .7299 for integer type.
- Manshadi et al also obtain a .823 inapproximation which implies the same (and current best known) inapproximation for the ROM model.

Extensions of online bipartite matching

- Vertex and edge weighted online matching
- Adwords (matching with offline budgets and edge weights)
- Applying the priority framework to matching problems
- Stochastic rewards matching (edges occur with probability)
- Online with Reassignments

The adwords problem: an extension of bipartite matching

- In the (single slot) adwords problem, the nodes in U are the online queries and the nodes in V are offline advertisers. For each query q and advertiser i, there is a bid $b_{q,i}$ representing the value of this query to the advertiser.
- Each advertiser also usually has a hard budget B_i which cannot be exceeded. The goal is to match the nodes in U to V so as to maximize the sum of the accepted bids without exceeding any budgets or to maximize profit when no bidder provides more profit than their budget. Without budgets and when each advertiser will pay for at most one query, the problem then is edge weighted bipartite matching.
- In the online case, when a query arrives, all the relevant bids are revealed.

Some results for the adwords problem

- Here we are just considering the combinatorial problem and ignoring game theoretic aspects of the problem.
- The problem has been studied for the special (but well motivated case) that all bids are small relative to the budgets. As such this problem is incomparable to the matching problem where all bids are in $\{0,1\}$ and all budgets are 1.
- For this small bid case, Mehta et al [2005) provide a deterministic online algorithm achieving the $1 - 1/e$ bound and show that this is optimal for all randomized online algorithms (i.e. adversarial input).

Greedy for a class of adwords problems

- Goel and Mehta [2008] define a class of adwords problems which include the case of small budgets, bipartite matching and b-matching (i.e. when all budgets are equal to some b and all bids are equal to 1).
- For this class of problems, they show that a deterministic greedy algorithm achieves the familiar $1 - 1/e$ bound in the ROM model. Namely, the algorithm assigns each query (.e. node in U) to the advertiser who values it most (truncating bids to keep them within budget and consistently breaking ties). Recall that Ranking can be viewed as greedy (with consistent tie breaking) in the ROM model.
- We note that some restriction has to be made for the edge weighted problem in the ROM model as this model generalizes the secretary problem for which a $1/e$ inapproximation is known.
- For the known i.i.d. integer type model, Haeupler et al [2011] give a .667 approximation and this is improved in Brubach et al [2016] to .705.

Vertex weighted bipartite matching for the adversarial worst case

- Aggarwal et al [2011] consider a vertex weighted version of the online bipartite matching problem. Namely, the offline vertices $v \in V$ all have a known weight w_v and the goal is now to maximize the weighted sum of matched vertices in V when again vertices in U arrive online.
- This problem can be shown to subsume the adwords problem when all bids $b_{q,i} = b_i$ from an advertiser are the same.
- It is easy to see that Ranking can be arbitrarily bad when there are arbitrary differences in the weight. Greedy (taking the maximum weight match) can be good in such cases. Can two such algorithms be somehow combined? Surprisingly, Aggarwal et al are able to achieve an adversarial worst casse 1-1/e approximation for this class of vertex weighted bipartite matching.
- The previously mentioned i.i.d. approximation resutls by Jaillet and Lu, and Brubach et al for bipartite matching were shown to apply to the vertex weighted case.

The vertex weighted online algorithm

The perturbed greedy algorithm

For each $v \in V$, pick r_v randomly in $[0, 1]$
Let $f(x) = 1 - e^{-(1-x)}$
When $u \in U$ arrives, match u to the unmatched v (if any) having the highest value of $w_v * f(x_v)$. Break ties consistently.

In the unweighted case when all w_v are identical this is the Ranking algorithm.

Some concluding remarks on max-sat and bipartite matching

- A research problem of current interest (work with Nicolas Pena) is to see to what extent some form of an extended online framework can yield a *deterministic* online bipartite matching algorithm with approximation ratio better than 1/2.
- As mentioned before, we can show that a 3/4 approximation for (submodular) max-sat can be obtained by a deterministic "poly width" online algorithm.
- One can formulate the Buchbinder and Feldman method in the framework of the priority BT model (pBT) of Alekhnovich et al. Can a bounded width online (or priority) BT algorithm obtain a 3/4 ratio?

Online and priority width inapproximations for max-sat and bipartite matching

We have the following width inapproximation results.

- To improve upon the $\frac{3}{4}$ max-sat approximation (using online width $2n$) result, we need exponential width. More precisely,
  For any $\epsilon > 0$ there exists $\delta > 0$ such that, for $k < e^{\delta n}$, no online width-cut-k algorithm can achieve an asymptotic approximation ratio of $3/4 + \epsilon$ for unweighted exact max-2-sat with input model 2.
- For any $\epsilon > 0$ there exists $\delta > 0$ such that, for $k < e^{\delta n}$, no pBT width-cut-k algorithm can achieve an asymptotic approximation ratio of $21/22 + \epsilon$ for unweighted max-2-sat with input model 3.
- For any $\epsilon > 0$, no bounded width online algorithm can achieve a $\frac{1}{2} + \epsilon$ approximation for bipartite matching.
- For any $\epsilon > 0$, no priority algorithm can achieve a $\frac{1}{2} + \epsilon$ approximation for bipartite matching.

A possible candidate for a "small" width deterministic online bipartite matching

Here is an idea for a max-of-k (for any number k, but the interesting question is what approximation ratio can we achieve when $k \in n^{O(1)}$). Let $M_1, \ldots, M_k$ be the matchings of the algorithm. Define the load of an offline vertex as the number of matchings in which it has been used. The algorithm does the following: when processing an online vertex, try to balance out the loads of the available offline vertices as much as possible. The intuition behind this is that we want the offline vertex with minimum load to have as high a load as possible, so if the adversary chooses to never show this vertex again, the number of matchings that will not match the vertex is as low as possible. The way of balancing loads can be made more precise or it could be left as an arbitrary choice for the algorithm. Nonetheless, this algorithm is fairly efficient and it would be interesting to see whether an algorithm that maintains polynomially many candidate matchings constructed this way can achieve an approximation ratio greater than $1/2$.

Some additional thoughts relating to online algorithms

- In the online research area, there are various studies of "online algorithms with advice". There are two such models, one being that the online algorithm is initially given some small $\alpha(n)$ bits of advice bits, based on the entire input.
- While certain advice strings seem reasonable (e.g. if they can be computed easily), the framework here allows any advice string.
- Mikkelsen [2015] shows that no deterministic online algorithm with sublinear $o(n)$ advice can be substantially better than any online randomized algorithm without advice.
- It follows that we cannot obtain anything substantially better than a $(1 - \frac{1}{e})$ apprroximation for online unweighted bipartite matching with $o(n)$ advice.
- By a result of Bockenhauer [2011] there is an online algorithm using $O(\log n)$ advice that achieves a $(1 - \epsilon)(1 - \frac{1}{e})$ approximation for unweighted bipartite matching, thus matching what is the best possible approximation with $o(n)$ advice.

Additional online thoughts continued

- Any online r bit advice algorithm immediately implies a non-uniform max-of-2^r online algorithm.
- Our inapproroximation about the limitation of width bounded online BT algorithms and max-of-k online algorithms for bipartite matching can then be used to show that $\Omega(\log \log n)$ advice is needed to achieve an approximation better than $\frac{1}{2} + \epsilon$.
- This advice result improves upon an $\Omega(\log \log \log n)$ lower bound on advice in a somewhat more restricted setting.
- These resuls show why it will be hard to prove inapproximation results about poly width online algorithms for bipartite matching. And also why it is problematic to establish inapproximations for multi-pass online algorithms.

Random walks and the random algorithm for 2-Sat and k-Sat

- First, here is the idea of the deterministic polynomial time algorithm for 2-Sat: We can first eliminate all unit clauses. We then reduce the problem to the directed $s - t$ path problem. We view each clause $(x \vee y)$ in F as two directed edges $(\bar{x}, y)$ and $(\bar{y}, x)$ in a graph G_F whose nodes are all possible literals x and $\bar{x}$. Then the formula is satisfiable iff there does not exist a variable x such that there are paths from x to $\bar{x}$ and from $\bar{x}$ to x in G_F.
- There is also a randomized algorithm for 2-SAT (due to Papadimitriou [1991]) based on a random walk on the line graph with nodes $\{0, 1, , n\}$. 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. $C(G) \leq 2m(n-1)$.

- It follows that the 2-SAT random walk has expected time at most $2n^2$ to find a satisfying assignment in a satisfiable formula. 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). Or 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 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 $(\frac{1}{e})^c$.