Lecture Notes CS:5350 Design and Analysis of Algorithms
Lecture 14: Oct 10, 2019
Scribe: Christopher Jenkins

Abstract

In this lecture, we study and solve several example problems in order to re-enforce topics we have covered in previous lectures. These include: Luby's randomized algorithm for calculating Maximal Independent Set; translation of the combinatorial problem Minimum Dominating Set to an Integer Program (IP), construction of the dual Linear Program (LP) from the linear relaxation of this, and interpretation of the integer restriction of this dual LP as a combinatorial program; and reduction of a given problem to Max Flow.

1 Luby's algorithm for Maximal Independent Set

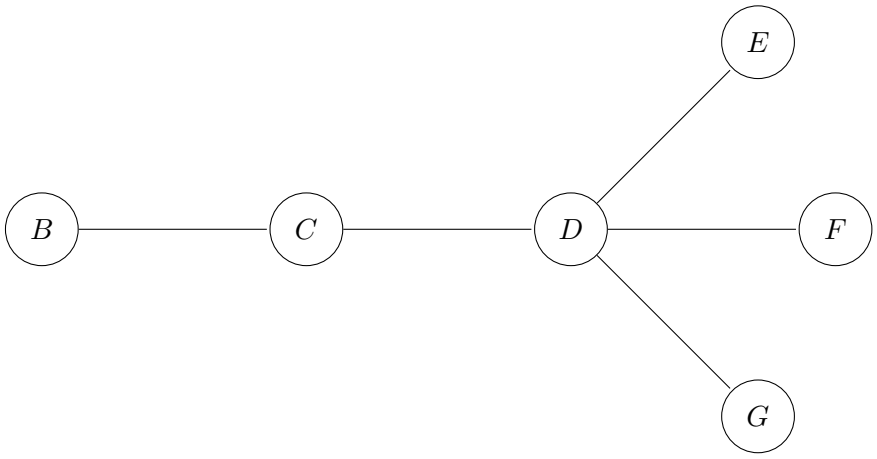

Figure 1: Practice problem for Luby's algorithm

Statement Suppose that Luby's MIS algorithm is executed on the graph given in Figure 1. Calculate the exact probability that vertex D becomes inactive (i.e., status = `ON` or `OFF`) after the execution of the first round of Luby's algorithm. Assume that vertex IDs are in alphabetical order,i.e., vertex B has ID 1, vertex C has ID 2, vertex D has ID 3 and so on.

Solution First, we know that the events of D being `ON` or D being `OFF` are disjoint, so the probability of D being either `ON` *or* `OFF` is the sum of the probabilities of these two events. We know that at the end of one round, the state of D is `ON` iff D marked itself and won tie-breaking with any node in its set of neighbors $\mathcal{N}(D)$ that also marked themselves. Furthermore, we know that the state of D is `OFF` iff some neighbor $v \in \mathcal{N}(D)$ marked itself and won tie-breaking with any of its neighbors in $\mathcal{N}(v)$. We have:

$$\begin{aligned}\mathcal{P}[D \text{ inactive}] &= \mathcal{P}[D \texttt{ ON} \vee D \texttt{ OFF}] \\ &= \mathcal{P}[D \texttt{ ON}] + \mathcal{P}[D \texttt{ OFF}] \\ &= \mathcal{P}[D \text{ marks itself} \wedge D \text{ wins all tie-breaking}] \\ &\quad + \mathcal{P}[D \text{ is unmarked} \wedge \exists v \in \mathcal{N}(D).(v \text{ marks itself} \wedge v \text{ wins all tie-breaking})]\end{aligned}$$

We consider first the left part of this sum. By a basic probability fact we have that $\mathcal{P}[D \text{ marks itself} \wedge D \text{ wins all tie-breaking}] = \mathcal{P}[D \text{ marks itself}] \cdot \mathcal{P}[\text{ wins all tiebreaking} \mid D \text{ marks itself}]$. The probability that D marks itself is given as $\frac{1}{2degree(D)} = 1/8$. The probability that D wins all tie-breaking that it enters, conditioned on the fact that it has marked itself, is 1 – recall that if two adjacent vertices mark themselves, ties are broken in favor of nodes with higher degree, and D has the highest degree of all its neighbors. So, the total probability in this case is just $1/8$.

Now we consider the right part of this sum. Using the same basic probability fact we have that this – the probability that 1) D is unmarked and 2) there is some neighbor v which marks itself and which wins all tie-breaking – is equal to the probability that D is unmarked times the probability that such a neighbor v exists, *conditioned* on the fact that D is unmarked.

$$\begin{aligned}&\mathcal{P}[D \text{ is unmarked} \wedge \exists v \in \mathcal{N}(D).(v \text{ marks itself} \wedge v \text{ wins all tie-breaking})] \\ = \;& \mathcal{P}[D \text{ is unmarked}] \\ &\cdot \mathcal{P}[\exists v \in \mathcal{N}(D).(v \text{ marks itself} \wedge v \text{ wins all tie-breaking}) \mid D \text{ is unmarked}]\end{aligned}$$

We consider first the left part of this product – it is simply the probability of the complement of the event D marks itself, given by $1 - 1/8 = 7/8$.

For the right part of this product, observe that $\mathcal{N}(d) = \{C, E, F, G\}$ and that we interpret the existentially quantification over this set as a union (logical *or*) of the given event for each neighbor of D. This is given as $\mathcal{P}[\bigvee_{v \in \{C,E,F,G\}} v \text{ marks itself} \wedge v \text{ wins tie-breaks} \mid D \text{ is unmarked}]$. Next, we use some intuition and observe that, on the condition that D is unmarked, every neighbor of D will *always* win any tie-breaking it enters into (E, F, G have no other neighbors with which to break ties, and C has a higher degree than its other neighbor B). So, the probability that $v \in \mathcal{N}(D)$ joins the MIS when D is unmarked reduces to the probability that v marks itself. Finally, to ease the calculation we can express this event by its complement (i.e., the conjunction over neighbors that *no* neighbor marks itself), and in turn express this as a product the probabilities of the individual neighbors (as marking events are independent).

$$
\begin{aligned}
& \mathcal{P}[\exists v \in \mathcal{N}(D).(v \text{ marks itself} \wedge v \text{ wins all tie-breaking}) \mid D \text{ is unmarked}] \\
= \; & \mathcal{P}[\bigvee_{v \in \{C,E,F,G\}} v \text{ marks itself} \wedge v \text{ wins tie-breaks} \mid D \text{ is unmarked}] \\
= \; & \mathcal{P}[\bigvee_{v \in \{C,E,F,G\}} v \text{ marks itself}] \\
= \; & 1 - \mathcal{P}[\bigwedge_{v \in \{C,E,F,G\}} v \text{ is unmarked}] \\
= \; & 1 - \left(\prod_{v \in \{C,E,F,G\}} \mathcal{P}[v \text{ is unmarked}] \right)
\end{aligned}
$$

Once we calculate probabilities for each of these events, the rest is straightforward calculation.

$$
\begin{aligned}
\mathcal{P}[C \text{ is unmarked}] &= 1 - \frac{1}{4} = \frac{3}{4} \\
\mathcal{P}[E \text{ is unmarked}] &= 1 - \frac{1}{2} = \frac{1}{2} \\
\mathcal{P}[F \text{ is unmarked}] &= 1 - \frac{1}{2} = \frac{1}{2} \\
\mathcal{P}[G \text{ is unmarked}] &= 1 - \frac{1}{2} = \frac{1}{2}
\end{aligned}
$$

The final probability is thus

$$\boxed{\frac{1}{8} + \frac{7}{8} \cdot (1 - \frac{3}{2^5})}$$

2 LP Duality and Minimum Dominating Set

Consider the definition of the Minimum Dominating Set problem:

Input: A graph $G = (V, E)$

Output: A set $S \subseteq V$ of smallest size such that for every vertex $v \in V$, either $v \in S$ or v is connected by an edge to some $v' \in S$

Figure 2 gives an example of a particular solution to this problem.

2.1 Primal IP (A)

For this exercise, we start by modeling this combinatorial problem as an integer program.

- We are seeking to choose among different vertices, so our choice variables are $x_v \in \{0, 1\}$ for each $v \in V$.

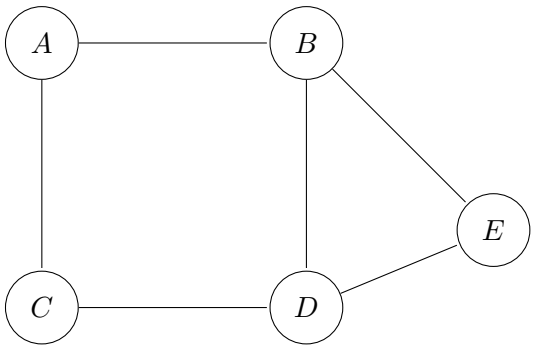

Figure 2: Example: $\{B, C\}$ forms a minimum dominating set

- We are trying to *minimize* the number of vertices selected, so this is a minimization IP.
- The constraint we must satisfy is that for every vertex v, there must be at least one vertex $w \in \mathcal{N}^+(v)$ (where $\mathcal{N}^+(v)$ is the set containing v and all vertices that are neighbors of v) such that w is chosen. Another way to say this is that the sum of all such x_w is no less than 1.

More formally, we have

$$\begin{aligned} \text{Minimize} \quad & \sum_{v \in V} x_v & & \\ \text{Subject to} \quad & \sum_{w \in \mathcal{N}^+(v)} x_w \geq 1 & & \text{for all } v \in V \\ & x_v \in \{0, 1\} & & \text{for all } v \in V \end{aligned}$$

2.2 Primal LP (B)

What happens when we consider relaxing the integrality constraints on IP (A) to create LP (B)? For the IP representing the MaxFlow problem, we observed that such relaxation does not change the optimal valuation of the objective function. In general, that situation does not hold for combinatorial problems; in particular, observe that for the graph in Figure 2 the solution $x_A = 1/2, x_B = 1/4, x_C = 1/4, x_D = 1/2, x_E = 1/4$ is feasible and evaluates to $1\frac{3}{4}$, which is better that the optimal solution 2 for IP (A) on the same graph.

2.3 Dual LP (C)

We will now construct the dual of LP (B) to produce LP (C). This process is essentially syntactic, in that we do not need to know much about the meanings of either LP (B) or LP (C) in order to carry it out.

- LP (B) is a minimization problem, ergo LP (C) is a maximization problem.
- For every choice variable x_v of LP (B), there ought to be a corresponding constraint in LP (C). This means there is a constraint in LP (C) for each $v \in V$.

- The constraints of LP (C) are bounded above by the coefficients attached to each choice variable x_v of the objective function of LP (B). In our case, this is just 1.
- For every constraint of LP (B), there ought to be a corresponding choice variable in LP (C). LP (B) has constraints for each vertex $v \in V$, so LP (C) has choice variables y_v for each $v \in V$.

At this point we have

$$\begin{aligned} \text{Maximize} \quad & \sum_{v \in V} y_v & & \\ \text{Subject to} \quad & ??? & \leq 1 \quad & \text{for all } v \in V \\ & y_v & \geq 0 \quad & \text{for all } v \in V \end{aligned}$$

The trickiest part of generating the dual of any LP is determining what ??? should be. We do this in general by constructing an incidence matrix where the rows are the constraints of the primal LP and the columns are the choice variables of the primal LP, and then re-interpreting this matrix with the *columns* the *constraints* of the dual LP and the *rows* the *choice variables* of the dual LP. Let us illustrate concretely with LP (B) and LP (C) instantiated to the graph given by Figure 2.

$$\begin{bmatrix} & A & B & C & D & E \\ A & 1 & 1 & 1 & 0 & 0 \\ B & 1 & 1 & 0 & 1 & 1 \\ C & 1 & 0 & 1 & 1 & 0 \\ D & 0 & 1 & 1 & 1 & 1 \\ E & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

We see already that the transpose of this matrix is the very same matrix. Interpreting the columns as constraints, we see for each vertex v there is an entry of 1 for every other vertex w it is adjacent to, and 0 otherwise. This is precisely the same expression we had before!

We conclude that the dual LP of (B) is

$$\begin{aligned} \text{Maximize} \quad & \sum_{v \in V} y_v & & \\ \text{Subject to} \quad & \sum_{w \in \mathcal{N}^+(v)} y_w & \leq 1 \quad & \text{for all } v \in V \\ & y_v & \geq 0 \quad & \text{for all } v \in V \end{aligned}$$

2.4 Dual IP (D)

Our last task is to give a combinatorial interpretation for IP (D), which is formed by replacing the positivity constraints of LP (C) with integrality constraints.

Input: A graph $G = (V, E)$

Output: A set $I \subseteq S$ of largest size such that for every vertex $v \in V$, at most one vertex in the neighborhood $\mathcal{N}^+(v)$ of v is in I.

In class we identified this as the MAXIMUM INDEPENDENT SET, but that is not correct. The additional constraint that vertices that are *not* in the selection cannot be adjacent to more than one vertex that *is* in the selection makes this a different problem, one whose name this scribe was unable to find.

3 MAXFLOW Reduction

Consider the following problem: some hospital is wishing to know whether it will be able to meet the demand for blood transfusions that it projects it will face in the next week. Recall that the four[1] blood types are A, B, AB, and O. Concerning blood transfusions, their relationships with each other are as follows:

- Patients with type A blood can receive transfusions of type A and type O blood
- Patients with type B blood can receive transfusions of type B and type O blood.
- Patients with type AB blood can receive transfusions of type A, B, AB, and O blood; such patients are called *universal receivers*
- Patients with type O blood can receive transfusions only of type O blood; donors with blood type O are called *universal donors*.

The hospital's current supply of each blood type is given by S_A, S_B, S_{AB}, and S_O, and its current projected demand is by patients of different blood types is D_A, D_B, D_{AB}, and D_O. The question is: how can we use an off-the-shelf MAXFLOW solver to determine whether there is some way for the hospital to meet its projected demand?

The flow network modeling this problem is given in Figure 3. For every flow modeling problem, we should ask ourselves: 1) what is the resource that we need to distribute; and 2) what are the constraints the problem places on it? The answer to the first question should suggest the left half of the constructed graph, and the answer to the second the right.

- Here, our four different blood types are our resource, modeled by the nodes A, B, AB, and O, and associated with each is the amount the hospital has available, represented by the outgoing edges of s with capacities S_A, S_B, S_{AB}, and S_O.

[1] Ignoring the positive / negative component

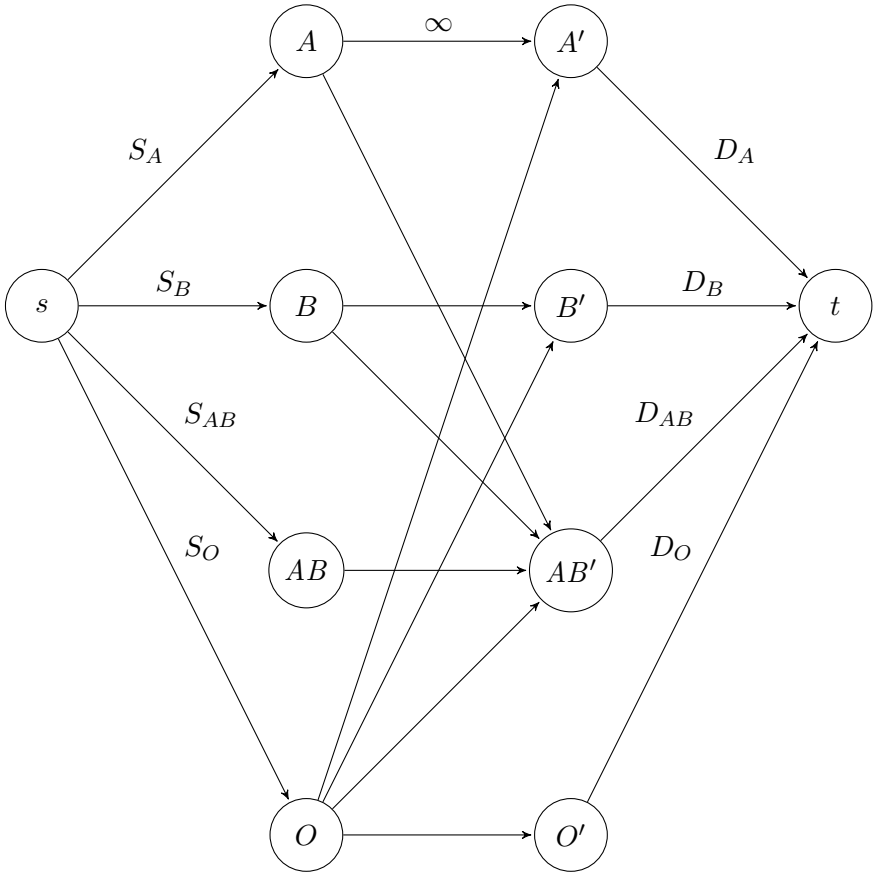

Figure 3: Network representing supply and demand of hospital blood supply

- The rules for which blood types can be given to which donors are represented by the presence or absence of edges between the nodes on the left of the graph and nodes A', B', AB', and O', which represent the four kinds of patients the hospital can treat.
- Edges from nodes A', B', AB', and O' to t are given capacities D_A, D_B, D_{AB}, and D_O, representing the projected demand for blood by each of the four different kinds of patients.
- Edges between the left hand side (A, B, AB, and O) and the right hand side (A', B', AB', and O') can be given capacities of ∞. This is because of flow conservation – the incoming edges of the nodes on the left have fixed capacities and so none of these can send infinite flow, and the outgoing edges of the nodes on the right also have fixed capacities and so none of these can receive infinite flow.

Having modeled the problem like so, we send the flow network (s, t, G, c) (where G is the graph, and c is the capacity function, described above) to a MaxFlow solver, receiving a flow f^*. If f^* saturates all of the incoming edges for t, then it follows that the hospital's projected demand can be satisfied by its current supply of blood; otherwise, lives could be lost.