Group assignment 2

Advanced Algorithms and Data Structures

Authors: psl788, wlc376, knx373

Hand-in deadline: December 6, 2022

CLRS 29.1-5

Bring the following Linear programming problem into slack form:

$$
\begin{array}{lrrrcl}
\text{Maximize} & 2x_1 & & -6x_3 & & \\
\text{Subject to} & x_1 & +x_2 & -x_3 & \leq & 7 \\
& 3x_1 & -x_2 & & \geq & 8 \\
& -x_1 & +2x_2 & +2x_3 & \geq & 0 \\
& & & x_1,\ x_2,\ x_3 & \geq & 0
\end{array}
$$

First step is to bring the problem to standard form by multiplying the the second and third constraint with -1. This gives:

$$
\begin{array}{lrrrcl}
\text{Maximize} & 2x_1 & & -6x_3 & & \\
\text{Subject to} & x_1 & +x_2 & -x_3 & \leq & 7 \\
& -3x_1 & x_2 & & \leq & -8 \\
& x_1 & -2x_2 & -2x_3 & \leq & 0 \\
& & & x_1,\ x_2,\ x_3 & \geq & 0
\end{array}
$$

To bring the standard form of this problem into slack form, the slack variables $x_4, x_5, x_6 \geq 0$ are introduced and the terms are rearranged such that the slack variables are on the LHS of the equalities. This gives the following slack form:

$$\begin{array}{rrrrr} z = & & 2x_1 & & -6x_3 \\ x_4 = & 7 & -x_1 & -x_2 & +x_3 \\ x_5 = & -8 & +3x_1 & -x_2 & \\ x_6 = & & -x_1 & +2x_2 & +2x_3 \end{array}$$

The basic variables are the variables on the LHS, x_4, x_5, x_6, and the non-basic variables are the variables on the RHS, x_1, x_2, x_3.

CLRS 29.2-6

The maximum-bipartite-matching problem can be expressed as a maximum flow problem as suggested in section 26.3 by defining the corresponding flow network $G' = (V', E')$ for the bipartite graph G.
Consider the case where $V = L \cup R$ is the vertex partition of G such that L and R are disjoint and all edges in E go between L and R. Let the source s and sink t be new vertices and define $V' = V \cup \{s, t\}$. The set of edges E' in the flow network are constructed by setting all edge capacities in E to one, and adding new directed edges of unit capacity from s to each vertex in L and from each vertex in R to t.
Much alike equations (29.47)-(29.50) in section 29.2, we can now write a linear program that given a bipartite graph $G = (V, E)$ solves the maximum-bipartite-matching problem:

$$\begin{array}{lll} \text{Maximize} & \displaystyle\sum_{v \in L} f_{sv} & \\ \text{Subject to} & f_{uv} \leq 1 & \text{for each } u, v \in V, \\ & \displaystyle\sum_{v \in V} f_{vu} = \sum_{v \in V} f_{uv} & \text{for each } u \in V - \{s, t\}, \\ & f_{uv} \geq 0 & \text{for each } u, v \in V. \end{array}$$

CLRS 29.3-5

Solve the following linear program using `simplex`:

$$\begin{array}{lrcrcl} \text{Maximize} & 18x_1 & + & 12.5x_2 & & \\ \text{Subject to} & x_1 & + & x_2 & \leq & 20 \\ & x_1 & & & \leq & 12 \\ & & & x_2 & \leq & 16 \\ & x_1, & x_2 & & \geq & 0 \end{array}$$

Writing the linear program in slack form, we obtain:

$$\begin{array}{lcllllll}
z & = & 0 & + & 18x_1 & + & 12.5x_2 \\
x_3 & = & 20 & - & x_1 & - & x_2 \\
x_4 & = & 12 & - & x_1 & & \\
x_5 & = & 16 & & & - & x_2 \\
x_1, x_2, x_3, x_4, x_5 & \geq & 0 & & & &
\end{array}$$

Initially, all nonbasic variables are set to zero such that we have $x_3 = 20$, $x_4 = 12$ and $x_5 = 16$. We start by investigating how much x_1 can be increased without violating any constraints. Recognising that x_4 is the binding constraint for x_1, it is apparent that x_1 can be increased by 12. By pivoting we now obtain:

$$\begin{array}{lcllllll}
z & = & 216 & - & 18x_4 & + & 12.5x_2 \\
x_3 & = & 8 & + & x_4 & - & x_2 \\
x_1 & = & 12 & - & x_4 & & \\
x_5 & = & 16 & & & - & x_2 \\
x_1, x_2, x_3, x_4, x_5 & \geq & 0 & & & &
\end{array}$$

Next, we investigate by how much x_2 can be increased. As x_3 is the binding constraint, we can see that x_2 can be increased by 8. Pivoting now results in:

$$\begin{array}{lcllllll}
z & = & 316 & - & 5.5x_4 & + & 12.5x_3 \\
x_2 & = & 8 & + & x_4 & - & x_3 \\
x_1 & = & 12 & - & x_4 & & \\
x_5 & = & 8 & - & x_4 & + & x_3 \\
x_1, x_2, x_3, x_4, x_5 & \geq & 0 & & & &
\end{array}$$

Since there are no more nonbasic variables in the objective function, the algorithm terminates. The solution we obtain is $(12, 8, 0, 0, 8)$ with an objective value of 316.

CLRS 29.4-1

Formulate the dual of the following (primal) linear programming problem:

$$\begin{array}{lrrcl}
\text{Maximize} & 18x_1 & +12.5x_2 & & \\
\text{Subject to} & x_1 & +x_2 & \leq & 20 \\
 & x_1 & & \leq & 12 \\
 & & +x_2 & \leq & 16 \\
 & & x_1,\ x_2 & \geq & 0
\end{array}$$

The dual can be obtained by identifying the constants used in the primal, see equation 29.16-18 in CLRS, and inserting these into equation 29.83-85 in CLRS. This gives the following dual linear programming problem:

$$
\begin{array}{llllll}
\text{Minimize} & 20y_1 & +12y_2 & +16y_3 & & \\
\text{Subject to} & y_1 & +y_2 & & \geq & 18 \\
 & y_1 & & +y_3 & \geq & 12.5 \\
 & & & y_1,\ y_2,\ y_3 & \geq & 0
\end{array}
$$

RandQS – expected bound on $\mathbb{E}[d(i)]$

Consider the proof for the number of comparisons performed by RandQS. In each iteration of the algorithm, the i'th smallest element is compared to the j'th smallest element as one of the two must be picked as a pivot. The conditional probability of picking i or j as pivot given that the pivot is picked uniformly at random in $\{S_{(i)}, S_{(i+1)}, \ldots, S_{(j)}\}$ is $p_{ij} = \dfrac{2}{j-i+1}$.

Suppose that the chosen pivot is j. Then we know that $d(i) > d(j)$, since i must be in a subtree of j. Thus, the expected depth of i is the expected number of ancestors. As we are only interested in the case where j is chosen as pivot, we have that $\Pr\big[\text{j is an ancestor of i}\big] = \dfrac{1}{j-i+1}$. Thus, we can now find the expected depth:

$$
\begin{aligned}
\mathbb{E}[d(i)] &= \sum_{j>i} \Pr\big[\text{j is an ancestor of i}\big] \\
&= \sum_{j>i} \frac{1}{j-i+1} \\
&\leq \sum_{k=1}^{n-i+1} \frac{1}{k} \\
&\leq \sum_{k=1}^{n} \frac{1}{k} \\
&= H_n = O(\log n).
\end{aligned}
$$

The last equality follows from Proposition B.4, as referred to in the randomized algorithms PDF.

Randomized contraction – 99% certainty of a min-cut

As per the last paragraph of page 8 of the PDF, *"The probability of discovering a particular min-cut [...] is larger than* $2/n^2$*"*.

Let m be the number of runs of the randomized contraction min-cut algorithm needed to achieve 99% or higher certainty of finding a minimum cut. Then m is the smallest integer satisfying the equation $m \cdot \frac{2}{n^2} \geq 0.99$:

$$\begin{aligned} & m \cdot \frac{2}{n^2} && \geq && 0.99 \\ \Leftrightarrow \quad & m && \geq && 0.495n^2. \end{aligned}$$

Since m is an integer and the RHS of above inequality is not always integral, the value we are looking for is $m = \lceil 0.495n^2 \rceil$.

Example: if $n = 4$, then $m = \lceil 7.92 \rceil = 8$ runs are required for 99% certainty of a min-cut, while for $n = 100$, $m = \lceil 4950 \rceil = 4950$ runs are required.

Randomized algorithms PDF – exercise 1.2

The general idea

We know that for any set S with $|S| = n$, there exist $m = 2^{n-1} - 1$ distinct ways to partition S into two non-empty subsets.

Consider a connected graph G with $|V| = n$. V is a set of distinct vertices and can also be partitioned in $m = 2^{n-1} - 1$ distinct ways.

The idea behind our solution is then this: Construct a graph G in such a way that the number of valid minimum cuts of G remain constant as n grows – if possible, then, since $m = 2^{n-1} - 1$ grows exponentially with n, we will have constructed a graph in which the ratio of valid minimum cuts to the total number of candidate cuts decreases exponentially with n.

Constructing the graph

Let $G = (V, E)$ be an undirected graph, with $V = \{v_0\} \cup V'$, such that V' is a non-empty clique [1] of $(n-1) \geq 1$ vertices in G, and let E, the set of edges,

[1] A subset of vertices in which every pair of two distinct vertices are adjacent.

consist of the set of edges in the clique V' as well as a single edge (v_0, v_i) for some vertex $v_i \in V'$. As such the size of V is:

$$\begin{aligned} |V| &= |V'| & + &\ |\{v_0\}| \\ &= (n-1) & + &\ 1 \\ &= n. \end{aligned}$$

Let $(v_0,\ v_1)$ with $v_1 \in V'$ be the single edge connecting the components V' and $\{v_0\}$. Then E is given by:

$$E = \Big\{(u, v) \in V^2\Big\} \bigcup \Big\{(v_0, v_1)\Big\}.$$

Figure fig. 1 shows an example graph for $n = 5$.

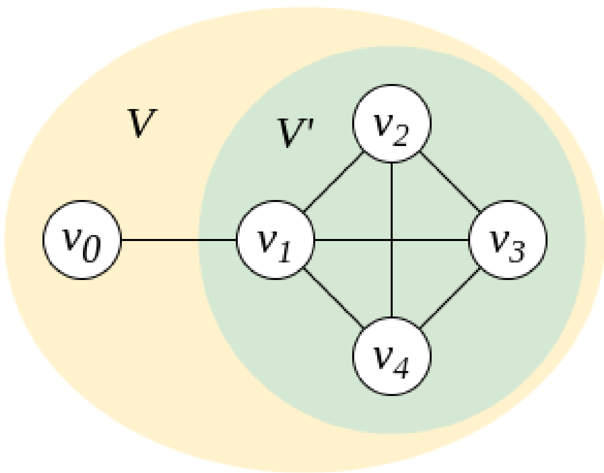

Figure 1: Example graph with $|V| = n = 5$ and $|V'| = 4$.

Clearly, for any graph G constructed in this way, the minimum cut is unique and is given by $C = \big(\{v_0\},\ V'\big)$. The cut-set [2] is $\big\{(v_0,\ v_1)\big\}$ and the value of the cut is always 1.

Since there are $m = 2^{n-1} - 1$ possible partitionings and only 1 valid cut, the probability of the algorithm randomly generating this partitioning is:

$$\frac{1}{2^{n-1} - 1} = \frac{1}{O(2^n)} = O\left(2^{-n}\right).$$

[2]The set of edges with one endpoint in either subset of the partition.

Randomized algorithms PDF – exercise 1.3

The pseudocode in fig. 2 shows how to obtain a Las Vegas algorithm from a Monte Carlo algorithm A. The function `to_las_vegas(A, pi)` takes as input a Monte Carlo algorithm `A` and a problem `pi` and repeatedly computes `solution = A(pi)` until a correct solution is produced. The pseudocode assumes an existing function `verify(pi, solution)` which, given a problem and a proposed solution, returns `True` if `solution` is a correct solution to the problem `pi`, and `False` otherwise.

```
1 function to_las_vegas(A, pi) {
2   do {
3     solution = A(pi);
4     success  = verify(pi, solution);
5   } while not success;
6
7   return solution;
8 }
```

Figure 2: Obtaining a Las Vegas algorithm from Monte Carlo algorithm A.

First, the `do-while` loop in lines 2-5 repeatedly executes `A(pi)` until a correct solution is found. The loop can possibly run indefinitely, but since `A` has probability of success $\gamma(n)$ and because the loop terminates immediately once a solution is verified as correct, we expect the loop to run for $\lceil 1/\gamma(n) \rceil$ iterations on average.

Secondly, since A has expected run time $O(T(n))$, line 3 of the loop runs in expected time $O(T(n))$, and because we assume that we can verify the solution in time $O(t(n))$, line 4 of the loop runs in expected time $O(t(n))$. In total, a single iteration of the `do-while` loop takes expected time $O(T(n) + t(n))$. Since we expect the loop to run for $\lceil 1/\gamma(n) \rceil$ iterations on average, the expected run time of the entire function is:

$$\begin{aligned} O\left(\left\lceil \frac{1}{\gamma(n)} \right\rceil \cdot (T(n) + t(n)) \right) &= O\left(\frac{1}{\gamma(n)} \cdot (T(n) + t(n)) \right) \\ &= O\left(\frac{T(n) + t(n)}{\gamma(n)} \right), \end{aligned}$$

which is what we wanted to show.

Summaries

psl788

Linear Programming

- Short introduction
- Standard and slack form
- Simplex algorithm
- Proof of weak duality

Randomized algorithms

- Random quicksort
  - Short explanation including an example
  - Proof of running time
- Las Vegas and Monte Carlo

wlc376 – exam presentation dispositions

Linear programming

1. intro:
2. standard vs slack form
3. give a small example of transforming a linear programming problem to standard, slack, and simplex form.
4. present SIMPLEX algorithm (and explain duality?)
5. prove weak duality

Randomized algorithms

1. intro and motivation: algorithms with random choices; for some types of programs, randomized algorithms are simpler, faster, or both.
2. Las Vegas vs Monte Carlo algorithms (RandQS vs edge contraction min-cut)
3. present RandQS (and why it is a good example of a randomized algorithm)

4. prove $O(n\log n)$ average case runtime of RandQS
5. if time: present edge contraction min-cut algorithm (??)

knx373

Linear programming and optimization

- Introduction to what a LP problem is. Keywords: Objective function, linear constraints, feasible solution/region, unbounded, optimal solution
- Standard and slack form
- Simplex algorithm
- Dual formulation of primal LP problem
- Weak duality including proof

Randomized algorithms

- Motivation for using randomized algorithms
- Las Vegas (random quick sort) vs Monte Carlo algorithms (random min cut)
- Itroduce random quick sort algorithm
- Proof of expected number of comparisons for RandQS
- If time: Converting algorithms (LV to MC and MC to LV)