MTAT.03.286: Advanced Methods in Algorithms **University of Tartu**

Homework Assignment 4 Solutions

---

1 Probabilistic algorithm

1. Let $S = \{x_1, x_2, \cdots, x_n\}$ be a set of Boolean variables of size $n \geq 1$.

   Definitions (reminder):

   - An *atom* is a Boolean variable from S. For example, x_3 is an atom (when $n \geq 3$).
   - A *literal* is an atom or its negation. For example, x_5 and $\neg x_2$ are two literals.
   - A *clause* is a disjunction of literals. For example, $(\neg x_1 \vee x_3 \vee \neg x_4)$ is a clause.
   - A *conjunctive normal form (CNF) formula* is a conjunction of some clauses. For example, $(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_3 \vee \neg x_4) \wedge (x_2 \vee x_{17})$ is a CNF formula.

   Let $\phi(x_1, x_2, \cdots, x_n)$ be a CNF formula with m clauses, where each clause contains exactly k different literals, and m and k are positive integers.

   (a) Propose a simple probabilistic algorithm that finds an assignment of True/False to variables in S, which satisfies the formula ϕ. The failure probability of the proposed algorithm should be at most $\frac{m}{2^k}$.
   Hint: if $\mathcal{E}_1, \mathcal{E}_2, \cdots, \mathcal{E}_m$ are random events, then (by the union bound):

   $$\mathsf{Prob}\left(\mathcal{E}_1 \cup \mathcal{E}_2 \cup \cdots \cup \mathcal{E}_m\right) \leq \sum_{i=1}^{m} \mathsf{Prob}(\mathcal{E}_i)\ .$$

   (b) How many parallel runs of the algorithm are needed in order to make the probability of a failure at most $\frac{1}{2^t}$, where $t \geq 1$ is a parameter?

   (c) Show an example of ϕ, which has no satisfying assignment for $m = 2^k$.

1.1 Algorithm

The algorithm has exactly one step:

- For each variable x_i assign a bit value uniformly randomly.

Report the obtained assignment as the satisfying assignment for ϕ.

To analyse the probability to that for each clause there is at most one assignment of variables where it is false. There can be clauses containing both literals x_i and $\neg x_i$ that are always true.

Each clause that does not contain pairs of x_i and $\neg x_i$ contains each variable exactly once, therefore there are k different variables involved. The algorithm picks each of these variables uniformly at random, therefore the probability that for a fixed clause the algorithm finds an assignment where the clause is false, is $\frac{1}{2^k}$. Let $\mathcal{E}_i$ be the event that clause i evaluates to False. The formula ϕ is False exactly if any of the events $\mathcal{E}_i$ happens and this corresponds to the case where the proposed algorithm returns a wrong result. We have $\mathsf{Prob}(\mathcal{E}_i) \leq \frac{1}{2^k}$ as $\mathsf{Prob}(\mathcal{E}_i) = 0$ if the clause contains both a variable and its negation.

Using union bound we get the following:

$$\mathsf{Prob}(error) = \mathsf{Prob}\left(\mathcal{E}_1 \cup \mathcal{E}_2 \cup \cdots \cup \mathcal{E}_m\right) \leq \sum_{i=1}^{m} \mathsf{Prob}(\mathcal{E}_i) \ \leq \sum_{i=1}^{m} \frac{1}{2^k} = \frac{m}{2^k};.$$

1.2 Parallel runs

Firstly, note that it is reasonable to consider improving the probability by parallel independent runs only if $\frac{m}{2^k} < 1$. Hence, we have that $m < 2^k$. Furthermore, each parallel instance is independent, therefore the probability that ℓ parallel instances all fail is $\left(\frac{m}{2^k}\right)^{\ell}$. Therefore, the question becomes to solve

$$\left(\frac{m}{2^k}\right)^{\ell} \leq \frac{1}{2^t} \ .$$

Which can be solved by taking a logarithm to obtain

$$\ell \geq \frac{t}{k - \log_2 m} \ .$$

1.3 Example of no satisfying assignment

For $m = 2^k$ we can start from the small examples, such as $k = 1$ and one variable where the formula $x \wedge \neq x$ is always false and we can generalize this. In any formula that has k variables and m distinct clauses that have each variable exactly once, they have no satisfying assignment. This is because there are 2^k different clauses of k variables (each time we either pick a variable or its negation into the clause) and then for each assignment of the variables exactly one clause will be false.

2 Maximum flow LP

Consider the following flow network $\mathcal{N}$.

1. Write the problem of finding maximum flow from s to t in $\mathcal{N}$ as a linear program. What is the value of the objective function?

2. Write down the dual of this linear program. There should be a dual variable for each edge of the network and for each vertex other than s and t.

   Now, consider a general flow network. Recall the linear program formulation for a general maximum flow problem, which was shown in the class.

3. Write down the dual of this general flow linear-programming problem, using a variable y_e for each edge and x_u for each vertex $u \neq s, t$.

4. Show that any solution to the general dual problem must satisfy the following property: for any directed path from s to t in the network, the sum of y_e values along the path must be at least 1.

2.1 Maximum flow

Let's use a variable f_e to denote the flow in each edge e.

The objective function is to maximize the incoming flow in vertex t and in this example it has no outgoing edges, therefore it is **max** $f_{b,t} - f_{t,a} - f_{t,s}$.

For each edge, we have to satisfy the edge rule, giving us the following constraints.

$$\begin{aligned} f_{s,a} &\leq 3 \\ f_{a,b} &\leq 6 \\ f_{b,t} &\leq 2 \\ f_{t,a} &\leq 4 \\ f_{t,s} &\leq 1 \\ f_{s,a} &\geq 0 \\ f_{a,b} &\geq 0 \\ f_{b,t} &\geq 0 \\ f_{t,a} &\geq 0 \\ f_{t,s} &\geq 0 \end{aligned}$$

In addition, the flow must satisfy the vertex rule for a and b.

$$\begin{aligned} f_{s,a} - f_{a,b} + f_{t,a} &= 0 \\ f_{a,b} - f_{b,t} &= 0 \end{aligned}$$

In total the problem becomes

$$
\begin{array}{lll}
\textbf{max} & f_{b,t} - f_{t,a} - f_{t,s} & \\
\textbf{s. t.} & f_{s,a} - f_{a,b} + f_{t,a} & = 0 \\
 & f_{a,b} - f_{b,t} & = 0 \\
 & f_{s,a} & \leq 3 \\
 & f_{a,b} & \leq 6 \\
 & f_{b,t} & \leq 2 \\
 & f_{t,a} & \leq 4 \\
 & f_{t,s} & \leq 1 \\
 & f_{s,a} & \geq 0 \\
 & f_{a,b} & \geq 0 \\
 & f_{b,t} & \geq 0 \\
 & f_{t,a} & \geq 0 \\
 & f_{t,s} & \geq 0
\end{array}
$$

The value of the maximum flow is 2 and we can find it either using any flow algorithms or by solving the linear program.

2.2 Dual problem

To find the dual problem, let's at first reformat the primal problem to make the matrix A more explicit. In addition, let's use variable x_v for each vertex rule of v and y_e for each edge rule on edge e.

$$
\begin{array}{lccccccc}
\textbf{max} & & & f_{b,t} & -f_{t,a} & -f_{t,s} & & \\
\textbf{s. t.} & f_{s,a} & -f_{a,b} & & +f_{t,a} & & = 0 & x_a \\
 & & f_{a,b} & -f_{b,t} & & & = 0 & x_b \\
 & f_{s,a} & & & & & \leq 3 & y_{s,a} \\
 & & f_{a,b} & & & & \leq 6 & y_{a,b} \\
 & & & f_{b,t} & & & \leq 2 & y_{b,t} \\
 & & & & f_{t,a} & & \leq 4 & y_{t,a} \\
 & & & & & f_{t,s} & \leq 1 & y_{t,s} \\
 & f_{s,a}, & f_{a,b}, & f_{b,t}, & f_{t,a}, & f_{t,s} & \geq 0 &
\end{array}
$$

It is straightforward to write out the dual problem using the columns for the matrix A that describes the constraints. In addition, all our initial variables have the non-negativity constraint meaning that all resulting constraints in the dual problem are inequalities. However, in the dual problem we only have non-negativity constraints for y_e variables for every edge e as these correspond to inequalities of the primal problem, but the x_v variables are unbounded.

$$
\begin{array}{rll}
\textbf{min} & 3y_{s,a} + 6y_{a,b} + 2y_{b,t} + 4y_{t,a} + y_{t,s} & \\
\textbf{s. t.} & x_a + y_{s,a} & \geq 0 \\
& -x_a + x_b + y_{a,b} & \geq 0 \\
& -x_b + y_{b,t} & \geq 1 \\
& x_a + y_{t,a} & \geq -1 \\
& y_{t,s} & \geq -1 \\
& y_{s,a} & \geq 0 \\
& y_{t,s} & \geq 0 \\
& y_{a,b} & \geq 0 \\
& y_{b,t} & \geq 0 \\
& y_{t,a} & \geq 0
\end{array}
$$

2.3 Dual of the general problem

The general linear program for a flow network $N(G(V,E),s,t,c)$ can be written out using a variable f_e for each $e \in E$ that denotes the flow in the corresponding edge.

$$
\begin{array}{lll}
\textbf{max} & \displaystyle\sum_{e \in in(t)} f_e - \sum_{e \in out(t)} f_e & \\
\textbf{s. t.} \ \forall e \in E: & & f_e \leq c(e) \\
\quad \forall v \in V \setminus \{s,t\}: & \displaystyle\sum_{e \in in(v)} f_e - \sum_{e \in out(v)} f_e = 0 & \\
\quad \forall e \in E: & & f_e \geq 0
\end{array}
$$

The initial problem has $|V \setminus \{s,t\}|$ equalities and $|E|$ inequalities if we do not count the basic restrictions for $f_e \geq 0$. Hence, the dual problem will have $|V \setminus \{s,t\}| + |E|$ variables. As we have the constraint $f_e \geq 0$ for all initial variables then we know that all the constraints in the dual problem will be inequalities. We use a variable y_e for each edge e and x_u for each vertex u that is not a source or the sink. Here x_u corresponds to the equality for the vertex rule of u and y_e corresponds to the inequality for the edge rule of e. We assume that there are no self loops because these are not interesting for maximum flow problems.

The objective function becomes minimizing

$$\sum_{e \in E} c(e) \cdot y_e \quad .$$

The constraints will be divided based on to which edge the derived inequality corresponds to.

We will get one constraint for each edge in the graph. The main difference between them is if the edge is connected to the source or sink or not.

For each edge that has one endpoint in the sink t we know that it was also used in the initial objective function, therefore the corresponding inequality will have the form $\ldots \geq 1$ or $\ldots \geq -1$. The rest of the equations will have the form $\ldots \geq 0$.

The most special edges are between s and t because they are not part of any vertex rule, but participate in the objective function. Therefore, we only have some rule for the edges as $y_{(s,t)} \geq 1$ and $y_{(t,s)} \geq -1$.

For each $e = (u,t) \in E$, $u \in V$ we have $-x_u + y_e \geq 1$. Because initially f_e is used in two constraints, one for the vertex rule of u and the other for the edge rule of e. In addition it is used in the objective function with a positive sign.

Analogously, for $e = (t,u) \in E$ we have $x_u + y_e \geq -1$ because this edge is used for the vertex rule of u with a positive sign and in the corresponding edge rule. However, it was present in the objective function with the coefficient -1.

Similarly, we get the special cases with $e = (s,u)$ for $u \in V \setminus \{t\}$ that gives us an inequality $x_u + y_e \geq 0$ because we use it in the edge rule and vertex rule for u and for $e = (u,s) \in E$ for $u \in V \setminus \{t\}$ that gives $-x_u + y_e \geq 0$.

Finally, we have the general edges $e = (u,v)$ where $u,v \in V \setminus \{t,s\}$. All these edges are reflected by three new variables and give $-x_u + x_v + y_e \geq 0$ because they are used in the vertex rules for u and v and for edge rule in e.

In addition, we have $y_e \geq 0$ because the variables for the edges correspond to the inequalities in the original problem.

Therefore, in total we obtain the following dual problem.

$$\begin{array}{lll}
\textbf{min} & & \displaystyle\sum_{e \in E} c(e) \cdot y_e \\
\textbf{s. t.} & \forall e = (t,s) \in E : & y_e \geq -1 \\
 & \forall e = (s,t) \in E : & y_e \geq 1 \\
 & \forall e = (t,u) \in E, u \in V \setminus \{s\} : & x_u + y_e \geq -1 \\
 & \forall e = (u,t) \in E, u \in V \setminus \{s\} : & -x_u + y_e \geq 1 \\
 & \forall e = (s,u) \in E, u \in V \setminus \{t\} : & x_u + y_e \geq 0 \\
 & \forall e = (u,s) \in E, u \in V \setminus \{t\} : & -x_u + y_e \geq 0 \\
 & \forall e = (u,v) \in E, u,v \in V \setminus \{t,s\} : & -x_u + x_v + y_e \geq 0 \\
 & \forall e \in E : & y_e \geq 0
\end{array}$$

2.4 Directed path

We know that none of the values y_e are negative, therefore if the value of one edge is larger than one, then this is sufficient for the proof.

The condition clearly holds for the only path with just one edge $y_{(s,t)}$ because we have the constraint $y_{(s,t)} \geq 1$.

Assume that we have some path using the numbered vertices $s - 1 - \ldots n - t$ such that the source and sink are passed through only once. Then we consider the sum $y_{(s,1)} + y_{(1,2)} + \ldots + y_{(n,t)}$. From the constraint $-x_n + y_{(n,t)} \geq 1$ we get $y_{(n,t)} \geq 1 + x_n$. Therefore

$$y_{(s,1)} + y_{(1,2)} + \ldots + y_{(n,t)} \geq y_{(s,1)} + y_{(1,2)} + \ldots + y_{(n-1,n)} + 1 + x_n \ \ .$$

Next, we can consider the constraints $-x_{n-1} + y_{(n-1,n)} + x_n \geq 0$ where $y_{n-1,n} + x_n \geq x_{n-1}$ gives

$$y_{(s,1)} + y_{(1,2)} + \ldots + y_{n-1,n} + x_n + 1 \geq y_{(s,1)} + y_{(1,2)} + \ldots + y_{(n-2,n-1)} + x_{n-1} + 1 \ \ .$$

We can keep using this rule on the intermediate edges (u, v) until we reach the sink and have only $y_{s,1} + x_1 + 1$ where we have $y_{s,1} + x_1 \geq 0$, therefore $y_{s,1} + x_1 + 1 \geq 1$. Following it back we get

$$y_{(s,1)} + y_{(1,2)} + \ldots + y_{(n,t)} \geq 1 \ \ .$$

We also have to consider the more general paths that pass through the source or the sink many times. Let $s - 1 - \ldots n - t$ be such path now where some $0 \leq i \leq n$ can be t or s.

We can consider this path in fragments that start and end in s or t. Each fragment is a path. Namely, we start from s and the first fragment ends when we first encounter t or s. We start the next fragment using this s or t and continue fragmenting until we reach the end of the path. The possible fragments are $s \to s$, $s \to t$, $t \to s$ and $t \to t$. However, from the previous we know that the sum is greater than one for each fragment $s \to t$. In addition, trivially each path from s to t has to contain at least one fragment $s \to t$ because otherwise we start from s and can only have fragments $s \to s$ which means that we never reach the t in the end of the path however we have to reach it by definition. The sum of y_e in each fragment is non-negative because each $y_e \geq 0$. Therefore, we can only look at the edges that belong to this fragment

$$\sum_{e \in \text{ path}} y_e \geq \sum_{e \in s \to t} y_e \geq 1$$

holds because the previous result for path from s to t clearly holds for this simple fragment $s \to t$.

3 Independent set and edge cover

Definition*. An independent set is a set of vertices in a graph, no two of which are adjacent. The maximum independent set problem is the problem of finding an independent set of maximum size.*

Definition*. An edge cover of a graph is a set of edges such that every vertex of the graph is incident with at least one edge of the set. The minimum edge cover problem is the problem of finding an edge cover of minimum size.*

Assume that we have an undirected graph $G(V, E)$.

3.1 Independent set as integer linear programming

For each vertex u we give a variable $x_u \in \{0, 1\}$. We have $x_u = 1$ if u is the in the independent set. Therefore the objective function becomes maximising

$$\sum_{u \in V} x_u$$

and we have a constraint for every edge only one of its endpoints can be in the independent set for $e = (u, v) \in E$ we have $x_u + x_v \leq 1$.

In total we get the following formalisation.

$$\begin{aligned} \textbf{max} \quad & \sum_{u \in V} x_u \\ \textbf{s. t.} \quad \forall e = (u, v) \in E : & x_u + x_v \leq 1 \\ \forall u \in V : \quad & x_u \in \{0, 1\} \end{aligned}$$

3.2 Minimum edge cover as integer programming

For each edge $e \in E$ we take a variable $y_e \in \{0, 1\}$ that is $y_e = 1$ if e is chosen to the edge cover. For a minimum edge cover we want to minimise

$$\sum_{e \in E} y_e \quad .$$

The constraint that each vertex $v \in V$ must be covered by at least one edge gives us the following constraint $\sum_{e \in inc(v)} y_e \geq 1$. Here $inc(v) = \{\{u, v\} : u \in V, \{u, v\} \in E\}$ is the set of all edges with one endpoint in v.

In total we get the following formalisation

$$\begin{aligned} \textbf{min} \quad & \sum_{e \in E} y_e \\ \textbf{s. t.} \quad \forall v \in V : & \sum_{e \in inc(v)} y_e \geq 1 \\ \forall e \in E : \quad & y_e \in \{0, 1\} \end{aligned}$$

3.3 Relax the two previous to general linear programming that are dual to each other

We want to relax the constraints $\forall u \in V : x_u \in \{0, 1\}$ and $\forall e \in E : y_e \in \{0, 1\}$ respectively.

In the independent set problem, we have $\forall u \in V : x_u \in \{0, 1\}$ and if we can at first relax it as $0 \leq x_u \leq 1$. In addition, we can relax it further to just the basic condition $0 \leq x_u$ because of the constraint $x_u + x_v \leq 1$ which can only be satisfied for non-negative variables if their values are upper bounded by 1. Therefore, we can drop the condition $x_u \leq 1$ because it is redundant in this problem. We get the following.

$$\begin{aligned}
&\textbf{max} && \sum_{u \in V} x_u \\
&\textbf{s. t.}\ \forall e = (u,v) \in E : && x_u + x_v \leq 1 \\
&\qquad \forall u \in V : && x_u \geq 0
\end{aligned}$$

For the minimum edge cover problem we have $\forall e \in E : y_e \in \{0,1\}$ which would be $0 \leq y_e \leq 1$ for general linear programming. Here we can actually drop the bound $y_e \leq 1$ because all variables y_e are used by the objective function and their coefficients are positive. Namely, assume that we have a optimum point with $y_e \geq 1$. Then we could decrease it to $y_e = 1$ which would decrease also the objective function meaning that it gives a new possible optimal solution. In addition, all the constraints would still be satisfied, because if $y_e = 1$ then all the sums $\sum_{e' \in inc(v)} y_{e'}$ that contain y_e would still have a value at least $\sum_{e' \in inc(v)} y_e \geq y_e = 1$. Therefore it is a contradiction the the assumption that we could have an optimal solution with $y_e \geq 1$. Hence, we obtain

$$\begin{aligned}
&\textbf{min} && \sum_{e \in E} y_e \\
&\textbf{s. t.}\ \forall v \in V : && \sum_{e \in inc(v)} y_e \geq 1 \\
&\qquad \forall e \in E : && y_e \geq 0
\end{aligned}$$

It is easy to see that by definition of the dual problem these problems are dual to each other. Consider the direction of finding the dual on the independent set problem. We have one one inequality for each edge, therefore the we would have new variables one for each edge. Let these be y_e. All the original variables x_u are used in the objective function with coefficient 1, therefore all new inequalities would have $\ldots \geq 1$. In addition, for each new variable $y_e \geq 0$ because all original constraints are inequalities. The final question is about the exact form of the new inequalities. We have one new inequality for each original variable, meaning one for each vertex. In addition, with respect to the new variables y_e it will be exactly a sum of all edges incident to this vertex, therefore $\forall v \in V$ we get $\sum_{e \in inc(v)} y_e \geq 1$. From this we also know that the other direction holds because a dual of the dual problem is the original problem.

4 Simplex algorithm

Solve the following linear-programming problem using simplex algorithm:

$$\begin{aligned}
&\textbf{max} && x_1 - x_2 + x_3 \\
&\textbf{s.t.} && x_1 + x_2 \leq 2 \\
& && 2x_1 + x_3 \leq 8 \\
& && x_2 + x_3 \leq 4 \\
& && x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0
\end{aligned}$$

4.1 Tableau format

We write down the initial problem in the tableau format and then use this to perform the simplex algorithm on this format. The first part is the matrix A representing the left hand side of the constraints. The middle part stands for the slack variables and the last column corresponds to the right hand side of the constraints $\vec{b}$. The last row holds the objective function information, especially it begins with $\vec{c}$.

$$\left(\begin{array}{ccc|ccc|c}
1 & 1 & 0 & 1 & 0 & 0 & 2 \\
2 & 0 & 1 & 0 & 1 & 0 & 8 \\
0 & 1 & 1 & 0 & 0 & 1 & 4 \\
\hline
1 & -1 & 1 & 0 & 0 & 0 & 0
\end{array}\right)$$

4.2 Steps of the simplex algorithm

The simplex algorithm in requires us to follow two rules and do Gaussian elimination.

1. Choose a column (variable) with a positive coefficient in the last row. If no such column exists then the current solution is optimal.
2. For each row i where this column c has a positive entry, compute the ratio $\frac{b[i]}{c[i]}$ between this entry and the right hand side of the corresponding constraint. Choose the row with the smallest ratio.

The chosen element is used as a pivot and then Gaussian elimination is performed.

First we can choose column 1 or 3. Let's choose 1 and the respective ratios are $\frac{2}{1}$ and $\frac{8}{2}$ meaning that we have to select row 1. We obtain the following matrix after applying the Gaussian elimination. (First row remains the same, Second row is $(2) - 2 \cdot (1)$, Third remains the same and Fourth row is (4)-(1)).

$$\left(\begin{array}{ccc|ccc|c}
1 & 1 & 0 & 1 & 0 & 0 & 2 \\
0 & -2 & 1 & -2 & 1 & 0 & 4 \\
0 & 1 & 1 & 0 & 0 & 1 & 4 \\
\hline
0 & -2 & 1 & -1 & 0 & 0 & -2
\end{array}\right)$$

Now the only column that we can choose is the third column and based on the respective ratios, we have to take either second or third row. We choose the third row. The first row remains the same, but we subtract row 3 from rows 4 and 2.

$$\left(\begin{array}{ccc|ccc|c}
1 & 1 & 0 & 1 & 0 & 0 & 2 \\
0 & -3 & 0 & -2 & 1 & -1 & 0 \\
0 & 1 & 1 & 0 & 0 & 1 & 4 \\
\hline
0 & -3 & 0 & -1 & 0 & -1 & -6
\end{array}\right)$$

In here we can not do any more steps as all variables have negative entries in the last row.

4.3 Result

The result of the Simplex algorithm can be written out from the final tableau based on the basic variables. Right now the basic variables are x_1, x_3 and z_2 as these are the variables with a single 1 in the respective columns. We can find the values for these variables in the last column of in the row where the respective columns have 1. Hence, the optimum is obtained at $x_1 = 2$ and $x_3 = 4$. We also know that $z_2 = 0$, but the value of the slack variable does not affect the objective function. In addition, the values of all non-basic variables is 0, hence $x_2 = 0$ and the value of the objective function is 6 (as represented in the bottom right entry in the tableau).