Homework 8

Collaborators: Jay Hou, Thomas Weng, David Marcano

Will Childs-Klein

wdc22

April 23, 2015

Problem 1

Algorithm

3CG-Min-Cost-Approx

$$
\begin{array}{l}
\textbf{\textit{input}}:\ G = (V, E, w), w : E \to \mathbb{R}^+ \\
\textbf{\textit{init}}:\ c = \emptyset, n = |v| \\
\textbf{\textit{begin}} \\
\quad \text{sort V such that } \sum_{(u,v_1) \in E} w(u, v_1) > \ldots > \sum_{(u,v_n) \in E} w(u, v_n) \\
\quad \textbf{\textit{for}}\ i = 1 \ldots n: \\
\quad\quad \text{let}\ N(i) = \{(i, u) \in E : u \in V\} \\
\quad\quad \textbf{\textit{for}}\ j \in \{1, 2, 3\}: \\
\quad\quad\quad \text{cost}(i, j) = \sum_{p \in N(i) : c(p) = j} w(p) \\
\quad\quad \textbf{\textit{end for}} \\
\quad\quad c(i) = \operatorname*{argmin}_{j \in \{1,2,3\}} \text{cost}(i, j) \\
\quad \textbf{\textit{end for}} \\
\quad \textbf{\textit{return}}\ \text{c} \\
\textbf{\textit{end}}
\end{array}
$$

Complexity

Theorem 8.1

3CG-Min-Cost-Approx is of complexity $O(n + n \log n)$

Proof of Complexity

lorem ipsum dolor sit amet

Correctness

Theorem 8.2

`3CG-Min-Cost-Approx` will alsays find a color assignment c whose total cost is at most

$$\frac{1}{3}\left(\sum_{(u,v)\in E} w(u,v)\right)$$

Proof of Theorem 8.1

Problem 2

Algorithm

3CG-Min-Cost-Randomized

```
input: G = (V, E, w), w : E → ℝ⁺
init:
begin
  let C be some constant
  let T = 1/3 Σ_{e∈E} w
  for i = 1...c:
    let c be a uniformly random assignment of c : v → {1, 2, 3}, ∀v ∈ V
    if cost(c) ≤ T
      return c
    endif
  endfor

  return c
end
```

Complexity

Correctness

- markov's inequality
- $C = 4$

Problem 3

Problem 4

Recall 3SAT

$$X = \{x_1, ..., x_n\}$$
$$C = \{C_1, ..., C_k\}$$
$$C_i = (x_j^i \vee x_h^i \vee x_l^i)$$

Where X is a list of n boolean variables, C is a list of k conjoined clauses of disjoint $x \in X$, and $1 \leq j, h, l \leq k$. In other words, each clause consists of at least 1 and at most 3 unique $x \in X$.

Integer Linear Program for 3SAT

When constructing our Integer Linear Program (ILP) to solve 3SAT, we can think of the process as similar to reducing a suspected NP-Complete problem to a known NP-Complete problem. Specifically, we will *transform* the inputs to 3SAT into inputs for our ILP, which we will call `3SAT-ILP`, such that the total number of clauses satisfied (by assignments for X found by `3SAT-ILP`) is maximized.

Let us refer to X^* as the assignment found by `3SAT-ILP` which is optimal in maximizing the number of clauses satisfied. The challenge in finding this "reduction" is twofold:

1. map 3SAT clause primitives (x_i,$\overline{x}_i$,`OR`) to linear combinations of
2. maximize the number of clauses satisfied

We will satisfy (`1.`) by representing `OR` as addition, x_i as x_i, and $\overline{x}_i$ as $(1 - x_i)$ in our linear combination to feed into `3SAT-ILP`. We will satisfy (`2.`) by adding a constraint vector c (lower-case, distinct from upper case C list of clauses) where $c = \{c_1, ..., c_k\}$. In other words, we will add a term c_j $(1 \leq j \leq k)$ to each equation in `3SAT-ILP`'s linear system constraint. The linear system constraint will be comprised of k distinct equations (one for each $C_j \in C$), each of four terms: 3 representing each $x \in X$ invoked in C_j, and the fourth being our c_j constraint. This c_j is what we will seek to minimize in our objective function. Take for example a 3SAT instance where $n = 4$, $k = 3$, and

$$C_1 = (x_1 \vee x_2 \vee \overline{x}_3)$$
$$C_2 = (x_1 \vee \overline{x}_2 \vee x_4)$$

$$C_3 = (x_2 \vee x_3 \vee \overline{x}_4)$$

Given this instance, we will construct `3SAT-ILP`'s linear inequality constraint system as follows:

$$\begin{aligned} x_1 + x_2 + (1 - x_3) + c_1 &\geq 1 \\ x_1 + (1 - x_2) + x_4 + c_2 &\geq 1 \\ x_2 + x_3 + (1 - x_4) + c_3 &\geq 1 \end{aligned}$$

through some simple algebra, we rearrange the above equations before inputting them into `3SAT-ILP` as follows:

$$\begin{aligned} c_1 \geq 1 - x_1 - x_2 - (1 - x_3) &\Leftrightarrow -x_1 - x_2 + x_3 \leq c_1 \\ c_2 \geq 1 - x_1 - (1 - x_2) - x_4 &\Leftrightarrow -x_1 + x_2 - x_4 \leq c_2 \\ c_3 \geq 1 - x_2 - x_3 - (1 - x_4) &\Leftrightarrow -x_2 - x_3 + x_4 \leq c_3 \end{aligned}$$

<NAME>

```
input:
init:
begin

end
```