CS 352 (Advanced Algorithms) **Winter 2012**
Problem Set #1 *Calder Coalson*
In collaboration with Daniel Alibi & David Anderson
With help from Carissa Knipe

1. (Daniel) See group submission.

2. (David) Our group came up with multiple algorithms and couldn't come to a consensus on which one was the most intuitive. I like this one though:

   Algorithm: Given a graph $G = \langle V, E \rangle$ and a partition $\langle S, V - S \rangle$:

   ```
   S := ∅
   forever
      if there exists a vertex in V − S with more uncut than cut edges
         add it to S
      else if there exists a vertex in S with more uncut than cut edges
         remove it from S
      else
         break
   return S
   ```

   Correctness: Each iteration of the loop increases the size of the cut by at least 1, so the algorithm will terminate. Furthermore, each vertex must have at least half its edges cut or the algorithm would not have terminated, so

   $$|\mathsf{Greedy}| \geq \frac{1}{2}|E| \geq \frac{1}{2}|\mathsf{OPT}|.$$

3. (Calder)

   (a) Consider the following input strings:

   $$S = \{1^{n+1}2^n,\ 2^n1^{n+1},\ 01^n\}.$$

   Greedy will immediately join the 1s of the first two strings, thereby eliminating the equally good (to within ϵ) join on the 2s, as well as a *second* equally good join to the third string. This leaves us with

   $$\begin{aligned}\lim_{n\to\infty} \frac{|\mathsf{Greedy}(S)|}{|\mathsf{OPT}(S)|} &= \lim_{n\to\infty} \frac{|01^n2^n1^{n+1}2^n|}{|01^{n+1}2^n1^{n+1}|} \\ &= \lim_{n\to\infty} \frac{4n+2}{3n+3} \\ &= 4/3.\end{aligned}$$

(b) Here are some things we thought about:

Sin: With every join, Greedy commits some measurable **sin** (not to be confused with sin). If S_i is the set of strings going into the ith iteration of Greedy's loop, then the sin of that iteration $\mathsf{sin}(i)$ is defined as

$$\mathsf{sin}(i) = |\mathsf{OPT}(S_{i+1})| - |\mathsf{OPT}(S_i)|.$$

Any join of size n commits at most n sin.

Potential Sin: The **potential sin** of an input string is

$$\mathsf{potsin}(s) = \min\left(\text{the largest potential join involving } s,\ \left\lfloor \frac{|s|}{2} \right\rfloor\right)$$

After greedy joins two strings s_1 and s_2 and commits n sin in the process, the resulting string s_{12} has potential sin

$$\mathsf{potsin}(s_{12}) = \min\left(\mathsf{potsin}(s_1), \mathsf{potsin}(s_2)\right) - n$$

Critical Characters: A prefix or suffix σ_i of string s_i is said to be **critical** if and only if $|\mathsf{OPT}(S)| > |\mathsf{OPT}(S')|$ where

$$S' = S \text{ with } \sigma_i \text{ removed from } s_i$$

Liabilities: We can think of every string $s = c_1c_2\cdots c_{k-1}c_k$, as a set of $2k+2$ segment-liability pairs where each pair is of the form

$$(c_1\cdots c_i, c_{i+1}\cdots c_k) \quad \text{or} \quad (c_{i+1}\cdots c_k, c_1\cdots c_i).$$

(c) Let $\Sigma = \{c_1, \ldots, c_k\}$ be the alphabet of the input set, and define the binary encoding of a given c_i as

$$\xi_i = 0^i(01)^{k+1-i}1^i.$$

Let σ_i be the binary encoding of the ith input string s_i, defined by concatenating the binary encodings of each character of s_i.

With this encoding, we want to show that σ and σ' will overlap by n encoded characters if and only if s and s' overlap by n real characters. Consider an arbitrary character $b(\xi_i) = 0^i(01)^{k+1-i}1^i$. This could potentially join a string at the beginning, end, or middle.

i. $0^i(01)^{k+1-i}1^i$ cannot join the end of another string, because no strings end with $0, 00, \ldots, 0^{i+1}$ or any other partial substring short of $0^i(01)^{k+1-i}1^i$. The only place you find $i+1$ consecutive 0s is followed by the rest of $b(\xi_i)$.

ii. It cannot join the beginning of another string, because no strings start with $1, 11, \ldots, 1^{i+1}$, or any other substring. Once again, the only place you find $i+1$ consecutive 1s is preceded by a complete $b(\xi_i)$.

iii. Since the $i+1$ zeros will *only* match other beginning-of-character zeros and the $(01)^{k+1-i}11$ and will only match other $b(\xi_i)$s, we know that $b(\xi_i)$ will only internally match itself.

Since each character can only overlap with itself, encoded strings will overlap only when their constituent characters do.

4. **Algorithm:** Given a graph $G = \langle V, E \rangle$ and an arbitrary ordering of vertices $\mathbf{V}$:

```
F := all forward edges in E (edges u → v such that u preceeds v in V
B := all backward edges in E (edges u → v such that u follows v in V
return the larger of F and B
```

Correctness: The only edges excluded from $F + B$ are length 1 cycles (edges $v \to v$) which cannot be in OPT. Thus $\mathsf{OPT} \subseteq F + B$, and therefore either

$$|F| \geq \frac{1}{2}|\mathsf{OPT}| \quad \text{or} \quad |B| \geq \frac{1}{2}|\mathsf{OPT}|.$$

5. **Algorithm:** Given a permutation $\pi = \langle \pi_1, \ldots, \pi_n \rangle$ of $\langle 1, \ldots, n \rangle$, define a **straight** Π_i to be the maximal consecutive sequence containing π_i which is already in the correct order.

   In other words, $\Pi_i = \langle \pi_s, \pi_{s+1}, \ldots, \pi_{e-1}, \pi_e \rangle$ such that:

   i. $s \leq i \leq e$
   ii. $\pi_s = \pi_j - (j - s)$ for all $s \leq j \leq e$
   iii. $\pi_{s-1} + 1 \neq \pi_i$ and $\pi_e + 1 \neq \pi_{e+1}$ (where $\pi_0 = 0$ and $\pi_{n+1} = n + 1$)

   Now consider the following algorithm:

```
return the number of unique straights in π
```

Correctness: Moving a single chunk can reduce the number of straights by at most 3, because a single move can:

   i. Join the two straights surrounding the moved chunk (-1 straights)
   ii. Join the moved chunk to its new left-neighbor (-1 straights)
   iii. Join the moved chunk to its new right-neighbor (-1 straights)

   Since any solution must reduce the number of straights to 1 and can do so at most 3 at a time, the number of straights is a 3-approximation to OPT.