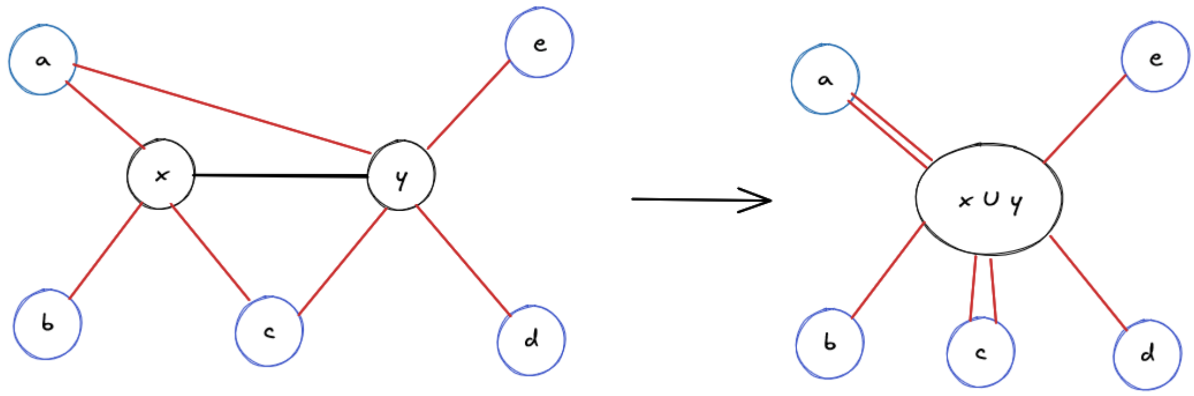
5. The Contraction Algorithm

> Un cut-set d'un graf no dirigit $G = (V, E)$ és un subconjunt d'arestes $C \subseteq E$ tals que si les esborrem d'E el graf resultant $(V, E-C)$ conté 2 o mésc omponents connexes. Un global min-cut o min-cut (depèn de les fonts bibliogràfiques) és un cut-set de cardinalitat mínima. Fixeu-vos que la cardinalitat d'un min-cut d'un graf G és el mínim nombre d'arestes que cal esborrar per a desconnectar G.
>
> Presenteu the Contraction Algorithm conegut també per Karger's Algorithm i analitzeu-ne eltemps de computació i la probabilitat d'error.

Algoritme

Durant $n-2$ passos, seleccionem una aresta (x, y) aleatòriament i en fusionem els extrems ($v_{\text{new}} = x \cup y$): l'aresta entre si desapareix i les que eren incidents a un o l'altre les reconnectem al vèrtex resultant de la fusió, v_{new}.



Considerem el graf com un multigraf (amb arestes repetides) i els vèrtexs com a conjunts (i.e., prenem $V = \{x_1, \ldots, x_n\}$ com a $V_0 = \{\{x_1\}, \ldots, \{x_n\}\}$). L'algoritme *contraction*, en pseudo-codi, és:

Input $G = (V_0, E_0)$:

> **for** $i \in [1, n-2]$:
>
> > $$(x, y) = \text{sample}(E_{i-1})$$
> >
> > $$v_{\text{new}} = x \cup y$$
> >
> > $$V_i = (V_{i-1} - \{x, y\}) \cup \{v_{\text{new}}\}$$
> >
> > $$E_i = (E_{i-1} - E_{\text{old}}) \cup E_{\text{new}}$$, on
> >
> > $$E_{\text{old}} = \text{incidents}(x) \cup \text{incidents}(y)$$
> >
> > $$E_{\text{new}} = \{(v_{\text{new}}, z) \mid z \in N(x) \cup N(y) \wedge z \neq x \wedge z \neq y\}$$
>
> **return** E_{n-2}

Anàlisi Probabilístic

Considerem el graf d'entrada $G = (V_0, E_0)$, que té un cut-set F amb cardinalitat $k = |F|$.

1. Considerem les variables aleatòries $X_i \equiv \text{sample}(E_{i-1}) \in F$ (i.e., que en la iteració i l'aresta aleatòria pertanyi al cut-set F) per tot $i \in [1, n-2]$.
2. L'algoritme retornarà el cut-set F (no un cut-set qualsevol, sinó exactament F) i només si si cap de les X_i és certa. Com que l'algoritme treu un vèrtex a cada iteració, al final quedaran dos vèrtexs S i T amb una aresta entre ells: les arestes entre S i T en el graf original defineixen el tall F. I.e., Si $\neg X_1 \cap ... \cap \neg X_{n-2}$ tenim que $E_{n-2} = \{(s,t) | s \in S \wedge t \in T\} = F$.
3. Per tant, la probabilitat que l'algoritme retorni F és:

$$\begin{aligned} \Pr[\text{èxit}] &= \Pr[\neg X_{n-2} \cap ... \cap \neg X_1] \\ &= \Pr[\neg X_{n-2} | \neg X_{n-3} \cap ... \cap \neg X_1] \cdot \Pr[\neg X_{n-2} \cap ... \cap \neg X_1] \\ &= \Pr[\neg X_{n-2} | \neg X_{n-3} \cap ... \cap \neg X_1] \cdot \Pr[\neg X_{n-2} | \neg X_{n-3} \cap ... \cap \neg X_1] \\ &\quad \cdot ... \cdot \Pr[\neg X_2 | \neg X_1] \cdot \Pr[\neg X_1] \end{aligned}$$

4. Considerem l'aresta (x, y) seleccionada en una iteració i: les arestes que tindrà el nou vèrtex $x \cup y$ seràn la unió de les que tenien x i y, menys (x, y).

   Per tant, $g(x \cup y) = g(x) - 1 + g(y) - 1$. A més, aquest valor coincideix amb la cardinalitat del tall $(\{x, y\}, V_i - \{x, y\})$.

   Podem extendre aquest raonament recursivament: per qualsevol $i \in [1, n-2]$, qualsevol $v \in V_i$, $g(v)$ coincideix amb el valor del tall $(v, V - v)$ en el graf original.

   Per tant, per tot $i \in [1, n-2]$, si $\neg X_1 \cap \ldots \cap \neg X_i$, llavors ***cap vèrtex*** $v \in V_i$ ***té grau menor que*** k (sinó $(v, V - v)$ tindria cardinalitat menor que k, absurd).
5. A cada iteració fusionem una aresta: convertim dos vèrtex en un de sol. Així, cada iteració ens deixa amb un vèrtex menys: $|V_i| = n - i$.
6. Pel *lema de les encaixades*

$$\sum_{v \in V_i} g(v) = 2|E_{i-1}| \geq k|V_i| = k(n-i) \iff |E_{i-1}| \geq \frac{k(n-i)}{2}$$

7. A cada iteració i, $\Pr[X_i | \neg X_{i-1} \cap ... \cap \neg X_1] = 1 - \Pr[\neg X_i | \neg X_{i-1} \cap ... \cap \neg X_1]$ és la probabilitat d'escollir una aresta de les k de F, d'entre les $|E_{i-1}|$ que queden. Per tant:

$$\begin{aligned} \Pr[\neg X_i | \neg X_{i-1} \cap ... \cap \neg X_1] &= 1 - \frac{k}{|E_{i-1}|} \geq 1 - \frac{k}{k(n-i)/2} \\ &= 1 - \frac{2}{n-i} = \frac{n-i-2}{n-i} \end{aligned}$$

8. Amb això i la definició del punt 3:

$$\begin{aligned} \Pr[\text{èxit}] &= \prod_{i=0}^{n-3} \frac{n-i-2}{n-i} = \frac{n-2}{n} \frac{n-3}{n-1} \frac{n-4}{n-2} \cdot ... \cdot \frac{3}{5} \cdot \frac{2}{4} \cdot \frac{1}{3} = (*) \\ &= \frac{1}{n} \frac{1}{n-1} \frac{n-2}{n-2} \cdot ... \cdot \frac{3}{3} \cdot \frac{2}{1} \cdot \frac{1}{1} = \frac{2}{n(n-1)} = 1 / \binom{n}{2} = \binom{n}{2}^{-1} \end{aligned}$$

   (*): *shiftem* tots els numeradors dues posicions a la dreta, afegint 1's on faci falta.
9. Si repetim l'algoritme per N intents independents, la probabilitat d'error serà

$$\Pr[\text{error}]^N = (1 - \Pr[\text{èxit}])^N = \left[1 - \binom{n}{2}^{-1}\right]^N$$

A més, sabem que $e^x \geq 1 + x$ per qualsevol valor de $x \geq 0$. Per exemple, ho podem veure amb la sèrie de Taylor:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + ... \geq 1 + x$$

Per tant, si apliquem aquesta desigualtat prenent $x = -\binom{n}{2}$:

$$\left[1 - \binom{n}{2}^{-1}\right]^N \leq \left[e^{-\binom{n}{2}^{-1}}\right]^N = e^{-N\binom{n}{2}^{-1}}$$

Si fem $N = \binom{n}{2}$ intents, tenim que la probabilitat d'error és $e^{-\binom{n}{2}\binom{n}{2}^{-1}} = e^{-1} \approx 0.37$.

Complexitat

A cada iteració recorrem les arestes de dos vèrtexs: com a molt $O(|V| = n)$. A més, fem $O(n)$ iteracions. Com que fem $\binom{n}{2} \in O(n^2)$ intents, el cost total és $O(n^4)$.

Conclusió

En resum, tenim un algoritme de cost polinomi amb probabilitat d'error $e^{-1} < \frac{1}{2}$: es tracta d'un algoritme probabilístic del tipus Monte-Carlo.