諸定義

定義 1. $f_1, ..., f_k$ を s-t 点素パスの集合とする。$U = (u_1, ...u_k)$ および $W = (w_1, ..., w_k)$ を $(f_1, ..., f_k)$ に関するスライスとし、すべての $1 \leq i \leq k$ について、u_i は f_i における w_i の前者または $u_i = w_i$ とする。この時 U は W より S に近いと言い、$U \preceq W$ と書く。もし $1 \leq i \leq k$ に対して、さらに $u_i \neq w_i$ ならば、U は W より厳密に s に近いと言い、$U \prec W$ と書く。同様に、W は U より厳密に t に近いと言う。便宜上、タプル $(s, s, ..., s)$ と $(t, t, ..., t)$ についても同様に上記を定義。したがって、例えば、$(s, s, ...s)$ はどのスライスよりも s に近いと言える。" $\preceq$ " は一般的な順序の合計を定義するものではない。

定義 2. U を任意のカットとする。s を含む $G[V \backslash U]$ の連結成分の頂点セットとして $V_s(U)$ を定義し、t を含む $G[V \backslash U]$ の連結成分の頂点セットとして $V_t(U)$ を定義し、V_r を $G[V \backslash U]$ の残りの連結成分の頂点集合の和集合として定義する。すなわち、s も t も含まないものである（したがって、$V_r(U)$ は空であり得る）。

定義 3. $(f_1, ..., f_k)$ に関して U をスライスとする。$U \preceq X$ であり、かつ $U \preceq X' \prec X$ を満たすカット X' が存在しないようなカットを X とする。この時 $U^+ := X$ を定義する。
上記のような X が存在しない場合は、$U^+ := (t, t, ..., t)$ とする。
同様に、$Y \preceq U$ かつ $U \preceq Y' \prec Y$ を満たすカット Y' が存在しないようなカットを Y とする。この時、$U^- := Y$ を定義する。
上記のような Y が存在しない場合は $U^- := (s, s, ..., s)$ とする

定義 4 (Subgraph Aggregation)**.** $G = (V, E)$ をネットワークグラフとし、$\mathcal{P} = (P_1, ...P_{|\mathcal{P}|})$ を part の集合とし、各 P_i について H_i を P_i のノード上の G の連結部分グラフとする。必ずしもグラフ $G[P_i]$ から誘導されるとは限らない。各部分グラフ H_i について、$V(H_i)$ 内のすべてのノードが部分グラフ H_i 内の隣接ノードを認識し、それ以外は何も知らないと仮定する。すべてのノード $v \in \bigcup_i P_i$ が O(log n) ビットの整数 x_v を持ち、$\oplus$ を長さ O（log n）の整数に作用する結合関数とする。P_i 内の各ノードは値 $\bigoplus_{v \in P_i} x_v$、すなわち P_i 内のすべての値 x_v の集合 $\oplus$ を知りたいとする。このようなタスクをオペレーター $\oplus$ における Subgraph Aggregation と呼ぶ。

アルゴリズム 1

Algorithm 1 U^+ の計算

初期設定:

点素 s-t パス $f_1, ..., f_k$

スライス $U = (u_1, ..., u_k)$

集合 $X = \{\}, x \in X$

$w_i := u_i$ ▷

for $i = 1 \text{to} k$ **do**
  w_i の前にあるノードをすべて X に入れる ▷ X は $V_s(U^+)$ の候補
end for
$Y := V \backslash (X \cup U), y \in Y$
while $\{x, y\} \in E$ がなくなるまで **do**
  if $y = t$ **then**
    return $(t, t, ..., t)$
  else if y が f_i に属している **then**
    X に f_i 上で y より前にある頂点を加える
    Y から X に加えた頂点を削除
    $w_i = y$
  else
    頂点 y を X にを入れて Y から削除する
  end if
end while
return $(w_1, ..., w_k)$

U^+ と U^- の計算時間は $O(m)$

分散案

1. 点素パス上の各頂点が隣接頂点にパス上のインデックスをブロードキャスト
2. 各点素パスの頂点を含まない G の連結成分内 (仮 C) で, 受け取ったパスのインデックスの最大値最小値を収集 (2kSA ラウンド)
3. スライス U のインデックスを各点素パス上のノードが知る (kSA ラウンド)
4. パス上の各頂点がスライス U のインデックスを隣接頂点にブロードキャスト (k ラウンド)
5. 各 C 内の頂点が点素パスの情報を 2 つ以上知っているとき
   - 各 C 内で受け取ったスライスのインデックスと大小比較
   - どれか一つでもスライスのインデックスより小さい時,C を X に追加
   - X 内でパスの最大値を収集 (kSA ラウンド)
6. loop 一番長い点素パスの長さ ℓ 分だけ繰り返す可能性
7. 収集したインデックスの最大値 (仮 w_i) の 1 つ前までのパスの頂点を X に追加
8. 最小値が x_i より小さい C を X に追加
9. 最大値収集,w_i を更新

ℓkSA ラウンド