Determinism in One-way Model: Finding Optimal Flows Efficiently

Mehdi Mhalla[1], Simon Perdrix[2]

[1]LIG, Université de Grenoble

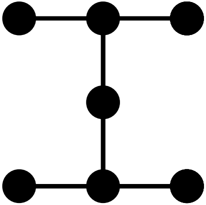
[2]Oxford University Computing Laboratory

LIP, 12 Nov. 2007

[quant-ph 0709.2670]

One-way model [Briegel - Raussendorf (00)]

One-qubit measurements over a large *entangled state* is universal for quantum computation.



Definition

For a given graph $G = (V, K)$,

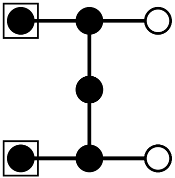
$$|G\rangle = \Pi_{(a,b)\in K} \Lambda Z_{a,b} |+\rangle_V$$

where $|+\rangle_V = \bigotimes_{u\in V} \frac{1}{\sqrt{2}}(|0\rangle_u + |1\rangle_u)$.

Definition (Open graph)

For any graph $G = (V, K)$, and for any $I, O \subseteq V$,

(G, I, O) is an open graph .



Definition (Initial state)

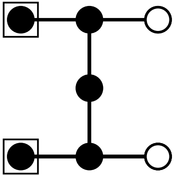
For any open graph (G, I, O) with $G = (V, K)$, for any $|I|$-qubit state $|\varphi\rangle$, the initial state is

$$\left|\Psi_{G,\varphi}\right\rangle = \left(\Pi_{(a,b)\in K}\Lambda Z_{a,b}\right)\left(|\varphi\rangle_I \otimes |+\rangle_{V\setminus I}\right)$$

Definition (Open graph)

For any graph $G = (V, K)$, and for any $I, O \subseteq V$,

(G, I, O) is an open graph .

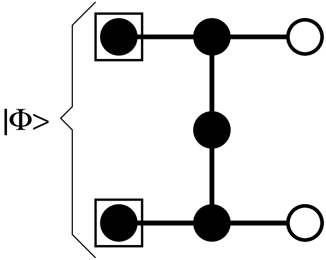


Definition (Initial state)

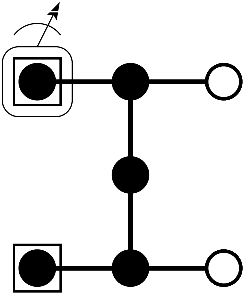
For any open graph (G, I, O) with $G = (V, K)$, for any $|I|$-qubit state $|\varphi\rangle$, the initial state is

$$\left|\Psi_{G,\varphi}\right\rangle = \left(\Pi_{(a,b)\in K} \Lambda Z_{a,b}\right) \left(|\varphi\rangle_I \otimes |+\rangle_{V\setminus I}\right)$$

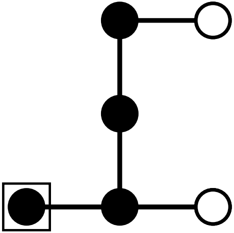
One-way quantum computation [Briegel, Raussendorf]



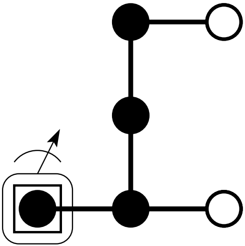
One-way quantum computation [Briegel, Raussendorf]



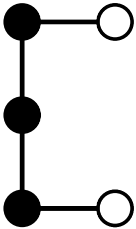
One-way quantum computation [Briegel, Raussendorf]



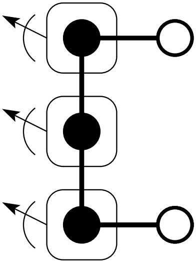
One-way quantum computation [Briegel, Raussendorf]



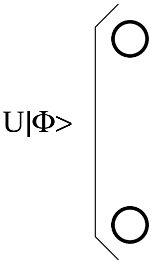
One-way quantum computation [Briegel, Raussendorf]

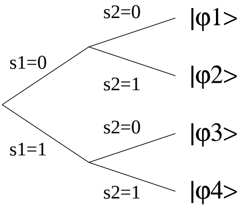


One-way quantum computation [Briegel, Raussendorf]



One-way quantum computation [Briegel, Raussendorf]





Definition (Determinism)

A One-way QC is **deterministic** iff all branches have the same output (up to a Pauli operator);

Theorem (Danos, Kashefi 05)

For a given (G,I,O), a deterministic One-way QC can be driven on the corresponding quantum state if (G,I,O) has a **causal flow**.

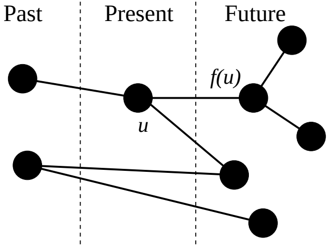
Definition (Causal Flow)

$(f,\prec)$ is a **causal flow** of (G,I,O), where $f : V\setminus O \to V\setminus I$, if for any u,

— $u \prec f(u)$

— $u \in N(f(u))$

— if $v \in N(f(u))\setminus\{u\}$ then $u \prec v$



Definition (Open Graph)

An **open graph** is a triplet (G,I,O), where $G=(V,E)$ is a undirected graph , and $I,O\subseteq V$, are respectively called input and output vertices.

Definition (Causal Flow)

$(f,\prec)$ is a **causal flow** of (G,I,O), where $f:V\setminus O\to V\setminus I$, if for any u,

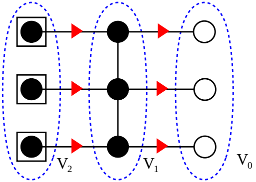
— $u\prec f(u)$

— $u\in N(f(u))$

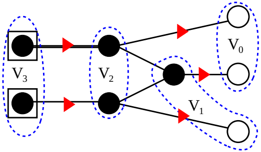
— if $v\in N(f(u))\setminus\{u\}$ then $u\prec v$

Definition (Layers and Depth)

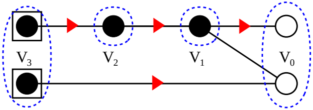
– $(f,\prec)$ induces a partition $(V_k^{\prec})_{k=0..d^{\prec}}$ of the vertices into $d^{\prec}+1$ **layers**, where $V_0^{\prec}=max(V)$ and $V_1^{\prec}=max(V\setminus V_0^{\prec})$, etc...
– $d^{\prec}$ is the **depth** of the flow.



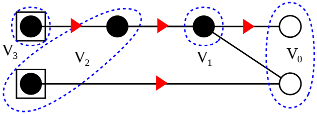
depth 2



depth 3



depth 3



depth 3

Theorem (De Beaudrap 06)

There exists a $O(nm)$[a] algorithm for finding causal flow of a given open graph (G,I,O), when $|I| = |O|$.

[a] where n (resp. m) is the number of vertices (resp. edges) of G

Open Question (Danos, Kashefi 05 & De Beaudrap 06)

Is there an efficient (poly-time) algorithm for finding a causal flow if $|I| \neq |O|$?

Theorem (Mhalla, Perdrix 07)

$O(m)$-algorithm for finding causal flow of a given open graph (G,I,O), whatever the numbers of inputs and outputs are.

Theorem (De Beaudrap 06)

There exists a $O(nm)^a$ *algorithm for finding causal flow of a given open graph* (G,I,O)*, when* $|I| = |O|$.

[a] where n (resp. m) is the number of vertices (resp. edges) of G

Open Question (Danos, Kashefi 05 & De Beaudrap 06)

Is there an efficient (poly-time) algorithm for finding a causal flow if $|I| \neq |O|$?

Theorem (Mhalla, Perdrix 07)

$O(m)$*-algorithm for finding causal flow of a given open graph* (G,I,O)*, whatever the numbers of inputs and outputs are.*

Theorem (De Beaudrap 06)

There exists a $O(nm)$[a] algorithm for finding causal flow of a given open graph (G,I,O), when $|I| = |O|$.

[a] where n (resp. m) is the number of vertices (resp. edges) of G

Open Question (Danos, Kashefi 05 & De Beaudrap 06)

Is there an efficient (poly-time) algorithm for finding a causal flow if $|I| \neq |O|$?

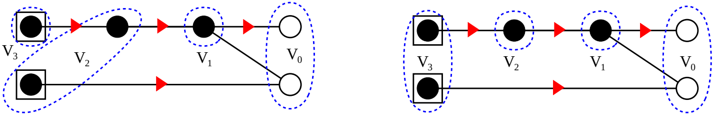
Theorem (Mhalla, Perdrix 07)

$O(m)$-algorithm for finding causal flow of a given open graph (G,I,O), whatever the numbers of inputs and outputs are.

Definition (Maximally Delayed)

$(f, \prec)$ is **maximally delayed** if for any causal flow $(f', \prec')$ of the same open graph,

$$\forall k, |\cup_{i=0..k} V_i^{\prec}| \geq |\cup_{i=0..k} V_i^{\prec'}|$$



Property

If $(f, \prec)$ *is a maximally delayed causal flow of* (G, I, O) *then*
$$V_0^{\prec} = O$$

Property

If $(f, \prec)$ *is a maximally delayed causal flow of* (G, I, O) *then*
$$V_1^{\prec} = \{u \in V \setminus V_0^{\prec}, \exists v \in V_0^{\prec}, N(v) \setminus V_0^{\prec} = \{u\}\}$$

Property (Inductive Structure)

If $(f, \prec)$ *is a maximally delayed causal flow of* (G, I, O) *then*
$$V_k^{\prec} = \{u \in V \setminus L_k, \exists v \in L_k, N(v) \setminus L_k = \{u\}\}$$
where $L_k = \bigcup_{i=0..k-1} V_i^{\prec}$

Property

If $(f,\prec)$ *is a maximally delayed causal flow of* (G,I,O) *then*

$$V_0^{\prec} = O$$

Property

If $(f,\prec)$ *is a maximally delayed causal flow of* (G,I,O) *then*

$$V_1^{\prec} = \{u \in V \setminus V_0^{\prec}, \exists v \in V_0^{\prec}, N(v) \setminus V_0^{\prec} = \{u\}\}$$

Property (Inductive Structure)

If $(f,\prec)$ *is a maximally delayed causal flow of* (G,I,O) *then*

$$V_k^{\prec} = \{u \in V \setminus L_k, \exists v \in L_k, N(v) \setminus L_k = \{u\}\}$$

where $L_k = \bigcup_{i=0..k-1} V_i^{\prec}$

Property

If $(f,\prec)$ *is a maximally delayed causal flow of* (G,I,O) *then*

$$V_0^{\prec} = O$$

Property

If $(f,\prec)$ *is a maximally delayed causal flow of* (G,I,O) *then*

$$V_1^{\prec} = \{u \in V \setminus V_0^{\prec}, \exists v \in V_0^{\prec}, N(v) \setminus V_0^{\prec} = \{u\}\}$$

Property (Inductive Structure)

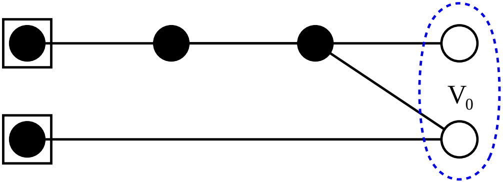
If $(f,\prec)$ *is a maximally delayed causal flow of* (G,I,O) *then*

$$V_k^{\prec} = \{u \in V \setminus L_k, \exists v \in L_k, N(v) \setminus L_k = \{u\}\}$$

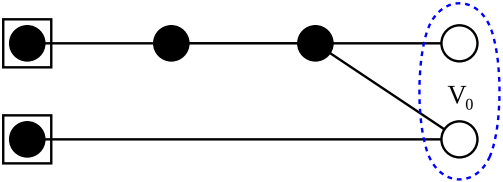
where $L_k = \bigcup_{i=0..k-1} V_i^{\prec}$

Greedy backward algorithm

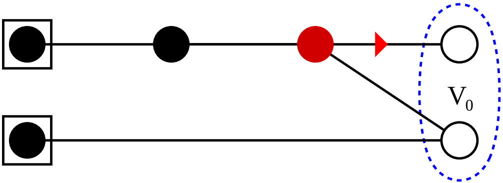
$$V_0^{\prec} := O$$



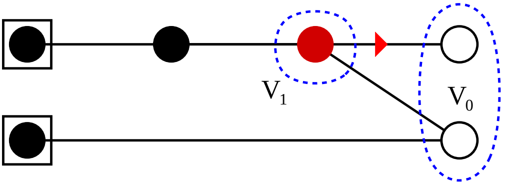
$$V_1^{\prec} := \{u \in V \setminus V_0^{\prec} \text{ s.t. } \exists v \in V_0^{\prec}, N(v) \setminus V_0^{\prec} = \{u\}\}$$



$$V_1^{\prec} := \{u \in V \setminus V_0^{\prec} \text{ s.t. } \exists v \in V_0^{\prec}, N(v) \setminus V_0^{\prec} = \{u\}\}$$



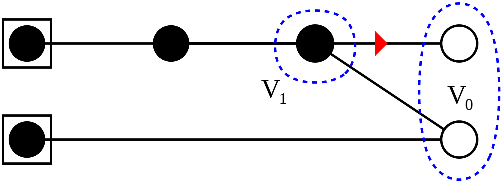
$$V_1^{\prec} := \{u \in V \setminus V_0^{\prec} \text{ s.t. } \exists v \in V_0^{\prec}, N(v) \setminus V_0^{\prec} = \{u\}\}$$



If $V_1^{\prec} \cup V_0^{\prec} = V$ then *'YES'*, if $V_1^{\prec} = \emptyset$ then *'NO'*

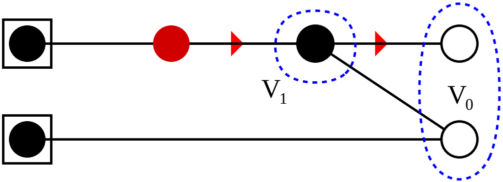
$$V_2^{\prec} := \{u \in V \setminus L_2 \text{ s.t. } \exists v \in L_2, N(v) \setminus L_2 = \{u\}\}$$

where $L_2 = V_1^{\prec} \cup V_0^{\prec}$



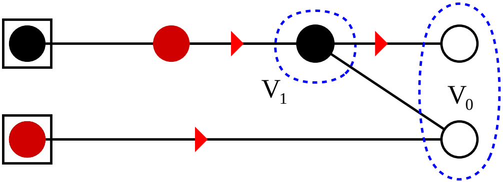
$$V_2^{\prec} := \{u \in V \setminus L_2 \text{ s.t. } \exists v \in L_2, N(v) \setminus L_2 = \{u\}\}$$

where $L_2 = V_1^{\prec} \cup V_0^{\prec}$



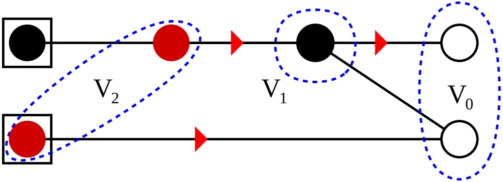
$$V_2^{\prec} := \{u \in V \setminus L_2 \text{ s.t. } \exists v \in L_2, N(v) \setminus L_2 = \{u\}\}$$

where $L_2 = V_1^{\prec} \cup V_0^{\prec}$



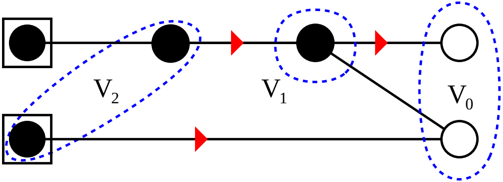
$$V_2^{\prec} := \{u \in V \setminus L_2 \text{ s.t. } \exists v \in L_2, N(v) \setminus L_2 = \{u\}\}$$

$$\text{where } L_2 = V_1^{\prec} \cup V_0^{\prec}$$



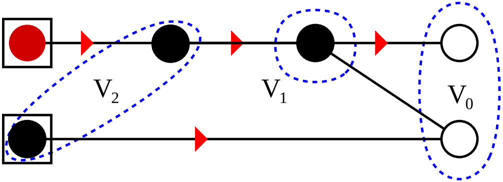
$$V_3^{\prec} := \{u \in V \setminus L_3 \text{ s.t. } \exists v \in L_3, N(v) \setminus L_3 = \{u\}\}$$

where $L_3 = V_2^{\prec} \cup V_1^{\prec} \cup V_0^{\prec}$



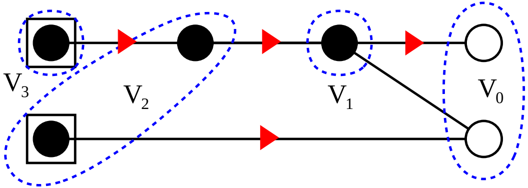
$$V_3^{\prec} := \{u \in V \setminus L_3 \text{ s.t. } \exists v \in L_3, N(v) \setminus L_3 = \{u\}\}$$

where $L_3 = V_2^{\prec} \cup V_1^{\prec} \cup V_0^{\prec}$



$$V_3^{\prec} := \{u \in V \setminus L_3 \text{ s.t. } \exists v \in L_3, N(v) \setminus L_3 = \{u\}\}$$

where $L_3 = V_2^{\prec} \cup V_1^{\prec} \cup V_0^{\prec}$



Greedy backward algorithm

For a given open graph (G,I,O)

$L := O$;
while $L \neq V$ **do**
  $C := \emptyset$
  for all $v \in L$ **do**
    if $|N(v) \setminus L| = 1$ **then** $C := C \cup (N(v) \setminus L)$;
  endfor
  if $C = \emptyset$ **then** 'no' ;
  $L := L \cup C$;
endwhile
'yes'

Correction

The produced flow $(f, \prec)$ is a causal flow of the input open graph (G, I, O).

Completeness

The algorithm produces a flow if the input open graph has a causal flow.

Complexity

$O(m)$ operations (model of adjacency list and adapted data structures)

Theorem (Optimal Flow)

The causal flow produced by the algorithm is **optimal***: there is no causal flow for the same open graph which has a smaller depth.*

Correction

The produced flow $(f, \prec)$ is a causal flow of the input open graph (G, I, O).

Completeness

The algorithm produces a flow if the input open graph has a causal flow.

Complexity

$O(m)$ operations (model of adjacency list and adapted data structures)

Theorem (Optimal Flow)

The causal flow produced by the algorithm is **optimal***: there is no causal flow for the same open graph which has a smaller depth.*

Correction

The produced flow $(f, \prec)$ is a causal flow of the input open graph (G, I, O).

Completeness

The algorithm produces a flow if the input open graph has a causal flow.

Complexity

$O(m)$ operations (model of adjacency list and adapted data structures)

Theorem (Optimal Flow)

The causal flow produced by the algorithm is **optimal**: *there is no causal flow for the same open graph which has a smaller depth.*

Correction

The produced flow $(f, \prec)$ is a causal flow of the input open graph (G, I, O).

Completeness

The algorithm produces a flow if the input open graph has a causal flow.

Complexity

$O(m)$ operations (model of adjacency list and adapted data structures)

Theorem (Optimal Flow)

The causal flow produced by the algorithm is **optimal**: *there is no causal flow for the same open graph which has a smaller depth.*

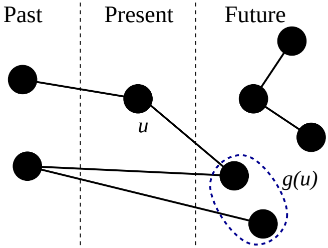
Definition (Generalized Flow)

$(g, \prec)$ is a **generalized flow** of (G, I, O), where $g : V \setminus O \to \wp(V \setminus I) \setminus \{\emptyset\}$, if for any u,

— if $v \in g(u)$, then $u \prec v$

— $u \in Odd(g(u)) = \{v \in V, |N(v) \cap g(u)| = 1[2]\}$

— if $v \in Odd(g(u)) \setminus \{u\}$ then $u \prec v$



Theorem (Perdrix 06)

Generalized flow is sufficient for determinism.

Theorem (Browne, Kashefi, Mhalla, Perdrix 07)

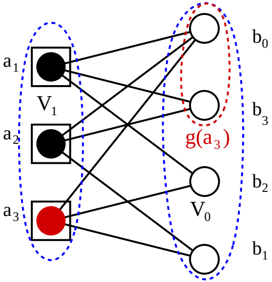
Generalized flow is necessary for uniform, strong and stepwise determinism.

Theorem (Perdrix 06)

Generalized flow is sufficient for determinism.

Theorem (Browne, Kashefi, Mhalla, Perdrix 07)

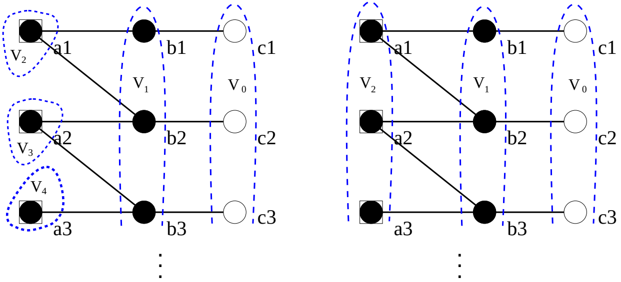
Generalized flow is necessary for uniform, strong and stepwise determinism.



$$\forall i = 1 \ldots 3, g(a_i) = \{b_0, b_i\}$$

Speed up

There exists an open graph of size $3n$ having an optimal causal flow of depth $n+1$ but a generalized flow of depth 2.



$f(a_i) = b_i$ and $f(b_i) = c_i$ $\qquad$ $g(a_i) = \{b_k\}_{k \le i}$ and $g(b_i) = \{c_i\}$

Open Question [Browne, Kashefi, Mhalla, Perdrix 06]

Is there an efficient (poly-time) algorithm for finding a generalized flow ?

Generalized Flow Algorithm [Mhalla, Perdrix 07]

$O(n^4)$ algorithm for finding generalized flow of a given open graph (G, I, O).

Open Question [Browne, Kashefi, Mhalla, Perdrix 06]

Is there an efficient (poly-time) algorithm for finding a generalized flow ?

Generalized Flow Algorithm [Mhalla, Perdrix 07]

$O(n^4)$ algorithm for finding generalized flow of a given open graph (G, I, O).

Property

If $(g, \prec)$ *is a maximally delayed generalized flow then* $V_0^{\prec} = O$.

Property

If $(g, \prec)$ *is a maximally delayed generalized flow then*

$$V_1^{\prec} = \{u \in V \setminus V_0^{\prec}, \exists X \subseteq V_0^{\prec}, Odd(X) \setminus V_0^{\prec} = \{u\}\}$$

Property (Inductive Structure)

If $(g, \prec)$ *is a maximally delayed generalized flow then*

$$V_k^{\prec} = \{u \in V \setminus L_k, \exists X \subseteq L_k, Odd(X) \setminus L_k = \{u\}\}$$

where $L_k = \bigcup_{i=0..k-1} V_i^{\prec}$

Property

If $(g, \prec)$ is a maximally delayed generalized flow then $V_0^{\prec} = O$.

Property

If $(g, \prec)$ is a maximally delayed generalized flow then

$$V_1^{\prec} = \{u \in V \setminus V_0^{\prec}, \exists X \subseteq V_0^{\prec}, Odd(X) \setminus V_0^{\prec} = \{u\}\}$$

Property (Inductive Structure)

If $(g, \prec)$ is a maximally delayed generalized flow then

$$V_k^{\prec} = \{u \in V \setminus L_k, \exists X \subseteq L_k, Odd(X) \setminus L_k = \{u\}\}$$

where $L_k = \bigcup_{i=0..k-1} V_i^{\prec}$

Property

If $(g, \prec)$ *is a maximally delayed generalized flow then* $V_0^{\prec} = O$.

Property

If $(g, \prec)$ *is a maximally delayed generalized flow then*

$$V_1^{\prec} = \{u \in V \setminus V_0^{\prec}, \exists X \subseteq V_0^{\prec}, Odd(X) \setminus V_0^{\prec} = \{u\}\}$$

Property (Inductive Structure)

If $(g, \prec)$ *is a maximally delayed generalized flow then*

$$V_k^{\prec} = \{u \in V \setminus L_k, \exists X \subseteq L_k, Odd(X) \setminus L_k = \{u\}\}$$

where $L_k = \bigcup_{i=0..k-1} V_i^{\prec}$

Greedy backward algorithm

For a given open graph (G, I, O)

$L := O$;
while $L \neq V$ **do**
  $C := \emptyset$
  for all $X \subseteq L$ **do**
    if $|Odd(X) \setminus L| = 1$ **then**
      $C := C \cup (Odd(X) \setminus L)$;
  endfor
  if $C = \emptyset$ **then** 'no' ;
  $L := L \cup C$;
endwhile
'yes'

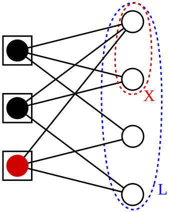
Greedy backward algorithm

For a given open graph (G, I, O)

$L := O$;
while $L \neq V$ **do**
  $C := \emptyset$
  for all $X \subseteq L$ **do**
    if $|Odd(X) \setminus L| = 1$ **then**
      $C := C \cup (Odd(X) \setminus L)$;
  endfor
  if $C = \emptyset$ **then** 'no' ;
  $L := L \cup C$;
endwhile
'yes'

$L := O$;
while $L \neq V$ **do**
  $C := \emptyset$
  for all $u \in V \setminus L$ **do**
    Solve $Odd(X) \setminus L = \{u\}$
    if there is a solution then
      $C := C \cup \{u\}$;
  endfor
  if $C = \emptyset$ **then** 'no' ;
  $L := L \cup C$;
endwhile
'yes'

Solve $Odd(X) \setminus L = \{u\}$



Definition (Cut-matrix)

$\Gamma_{(V\setminus L)\times L}$ is the $|V\setminus L| \times |L|$ sub adjacency matrix:

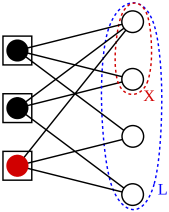
$$\Gamma = \left(\begin{array}{c|c} \cdot & \cdot \\ \hline \Gamma_{(V\setminus L)\times L} & \cdot \end{array}\right)$$

Linear System

$$Odd(X)\setminus L = \{u\} \iff \Gamma_{(V\setminus L)\times L}.\mathbb{I}_X^L = \mathbb{I}_{\{u\}}^{V\setminus L} \text{ in } \mathbb{F}_2$$

where $\mathbb{I}_X^A$ is $|A|$-vector such that $\forall u \in A, \mathbb{I}_X^A(u) = \begin{cases} 1 & u \in X \\ 0 & u \notin X \end{cases}$.

Solve $Odd(X) \setminus L = \{u\}$



Definition (Cut-matrix)

$\Gamma_{(V\setminus L)\times L}$ is the $|V \setminus L| \times |L|$ sub adjacency matrix:

$$\Gamma = \left(\begin{array}{c|c} \cdot & \cdot \\ \hline \Gamma_{(V\setminus L)\times L} & \cdot \end{array}\right)$$

Linear System

$$Odd(X) \setminus L = \{u\} \iff \Gamma_{(V\setminus L)\times L}.\mathbb{I}_X^L = \mathbb{I}_{\{u\}}^{V\setminus L} \text{ in } \mathbb{F}_2$$

where $\mathbb{I}_X^A$ is $|A|$-vector such that $\forall u \in A, \mathbb{I}_X^A(u) = \begin{cases} 1 & u \in X \\ 0 & u \notin X \end{cases}$.

Theorem (Optimal Flow)

The generalized flow produced by the algorithm is **optimal***: there is no generalized flow for the same open graph which has a smaller depth.*

1 Context and Definitions

2 Causal Flow Algorithm

3 Generalized Flow

4 Conclusion

- Deterministic One-way QC as the existence of a *causal flow* or a *generalized flow* on a graph
- Polytime algorithm [De Beaudrap 07] for finding causal flow if $|I| = |O|$ (I: input vertices, O: output vertices)
- Faster polytime algorithm for finding causal flow (even if $|I| \neq |O|$)
- Polytime algorithm for finding generalized flow.
- These last two algorithms produce *optimal flows*, minimizing the depth of the One-way QC.
- Automatic parallelisation of one-way QCs (or quantum circuits).