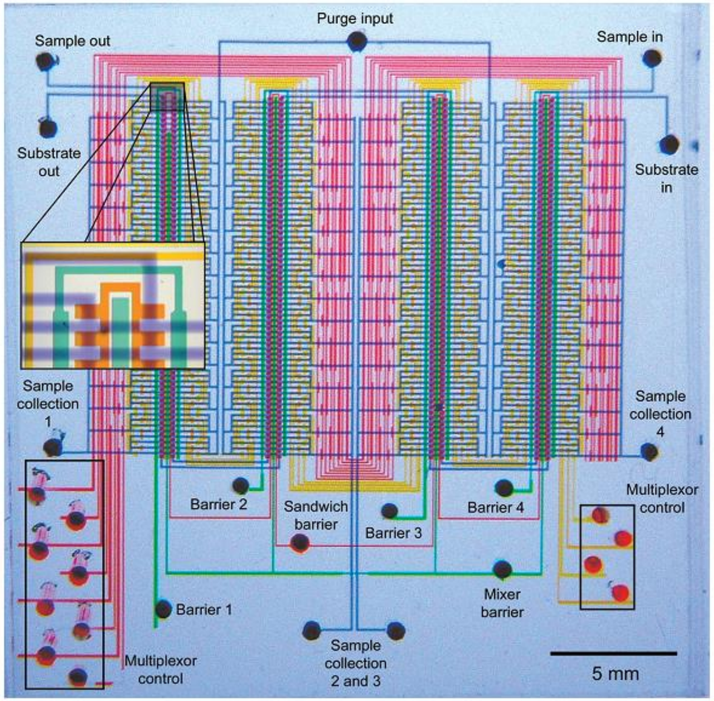
Wash Optimization for Cross-Contamination Removal in Flow-Based Microfluidic Biochips



Kai Hu[1], Tsung-Yi Ho[2]

Krishnendu Chakrabarty[1]

[1]Department of Electrical & Computer Engineering
Duke University
Durham, NC 27708, USA

[2]Department of Computer Science and Information Engineering
National Cheng-Kung University
Tainan, Taiwan

Duke University

Objectives

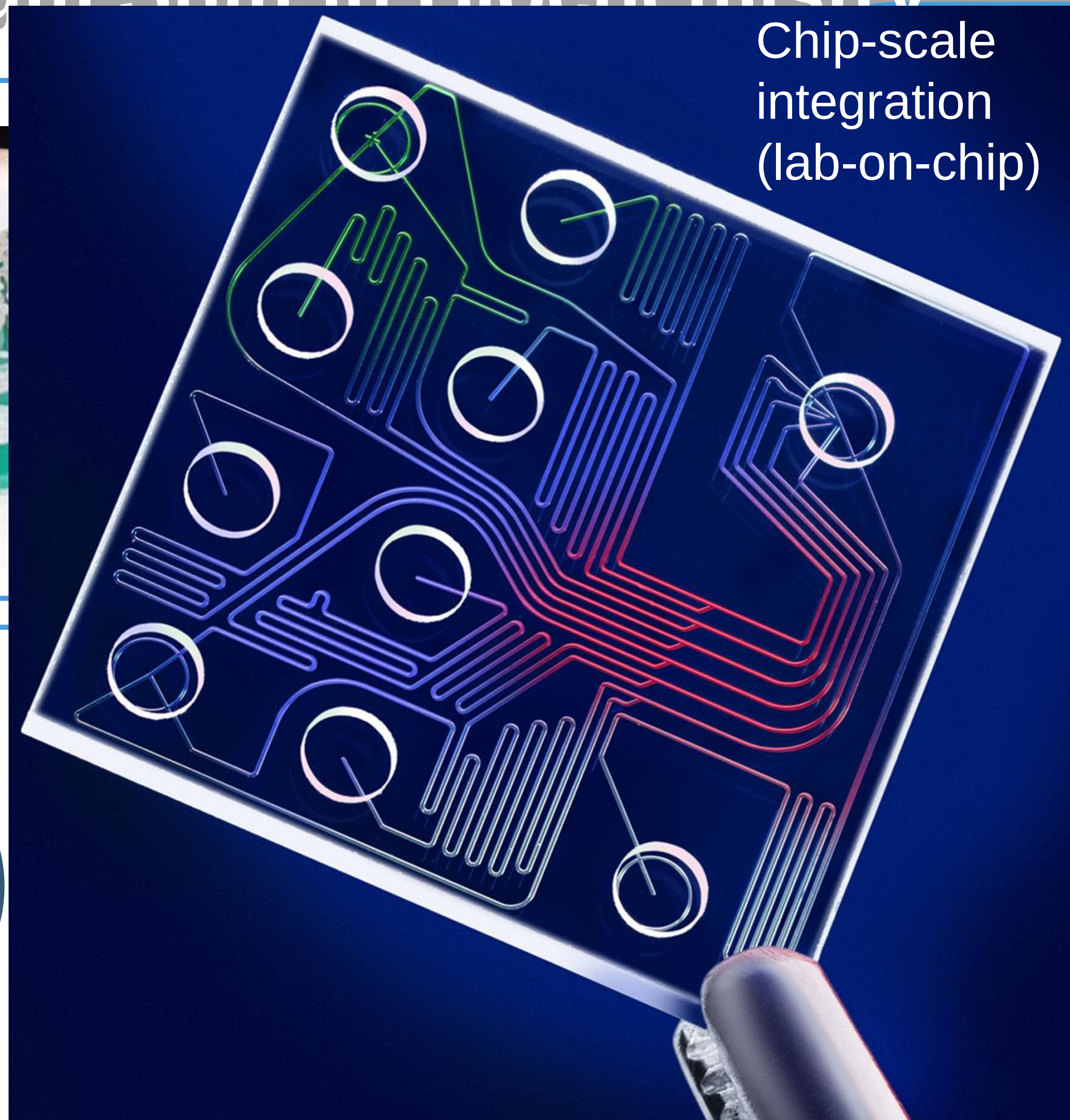
- Automated cross-contamination removal in flow-based microfluidic biochips
  - Wash microchannels by generating buffer flows that cover all wash targets.
  - Avoid interruption of other concurrent tasks
  - Wash time minimization

Outline

- Problem formulation
  - Discretized graph of continuous channels networks
  - Wash path implementability
  - Wash targets, occupied channels and wash time
- Identification of washing-path set
  - Generation of path dictionary
  - Storage of the Path dictionary
  - Weighted hitting-set problem
  - Utility function
- Results
  - Application to two fabricated biochips
- Conclusions

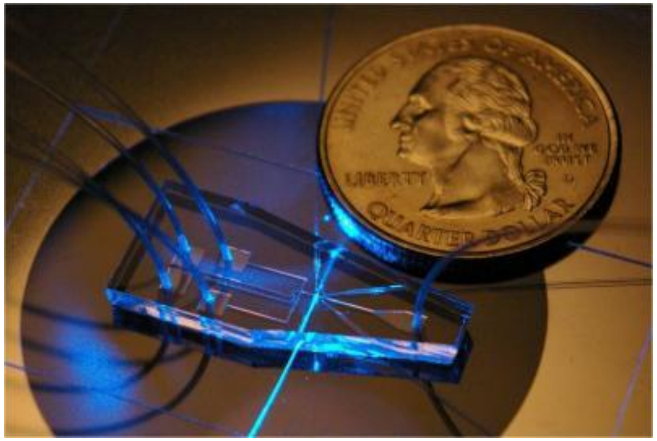
Paradigm Shift in Biochemistry

Skilled Technicians

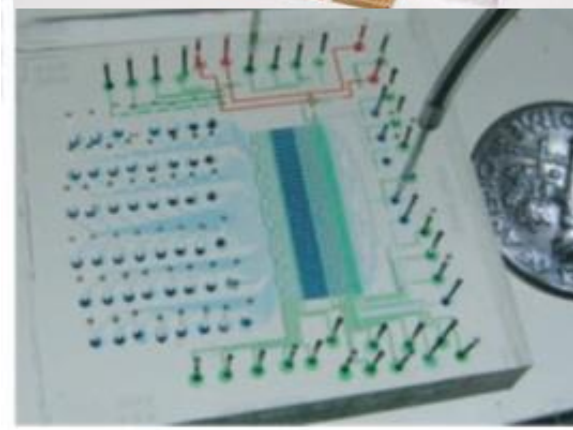


ulky
pment

Why Biochips?



Small

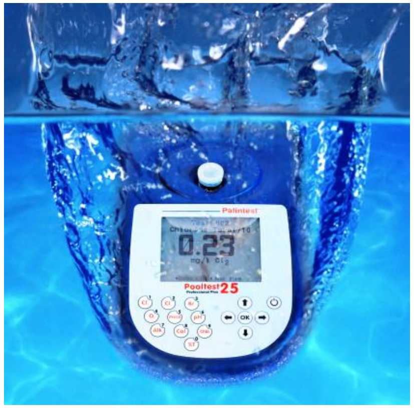


High Throughput

Automation

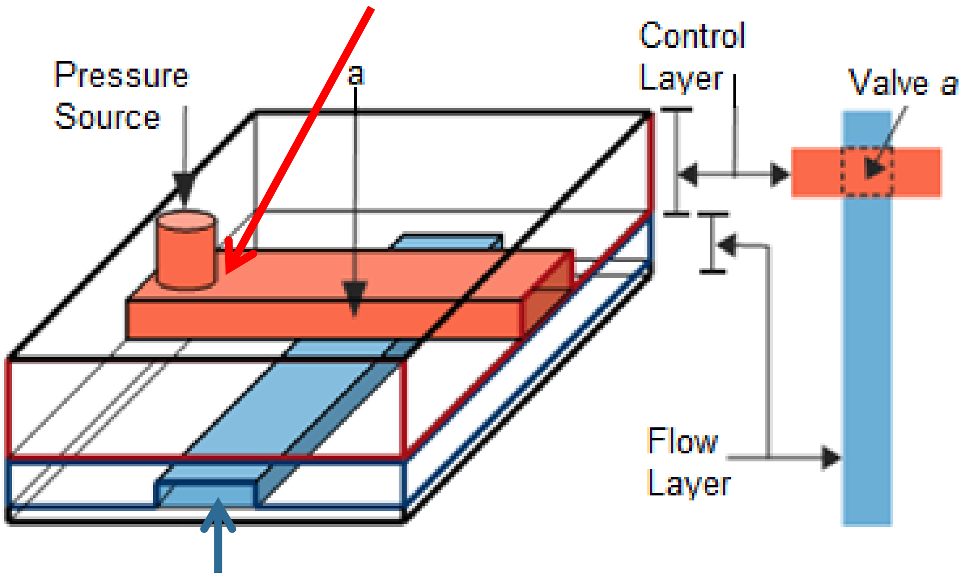
Applications

- Drug discovery
- Point-of-care devices
- Preventive individualized care
- Bio-hazard detection
- DNA sequencing



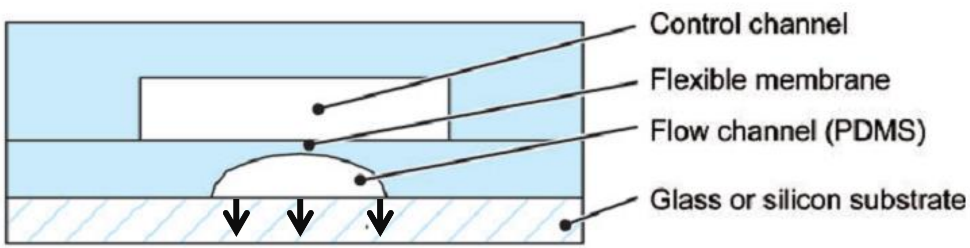
Flow-Based Microfluidics

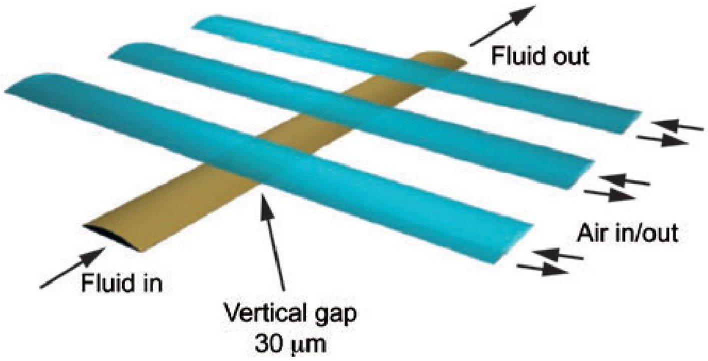
High pressure in the control channel



Valve: a flexible membrane at the overlapping area between channels of the two layers.

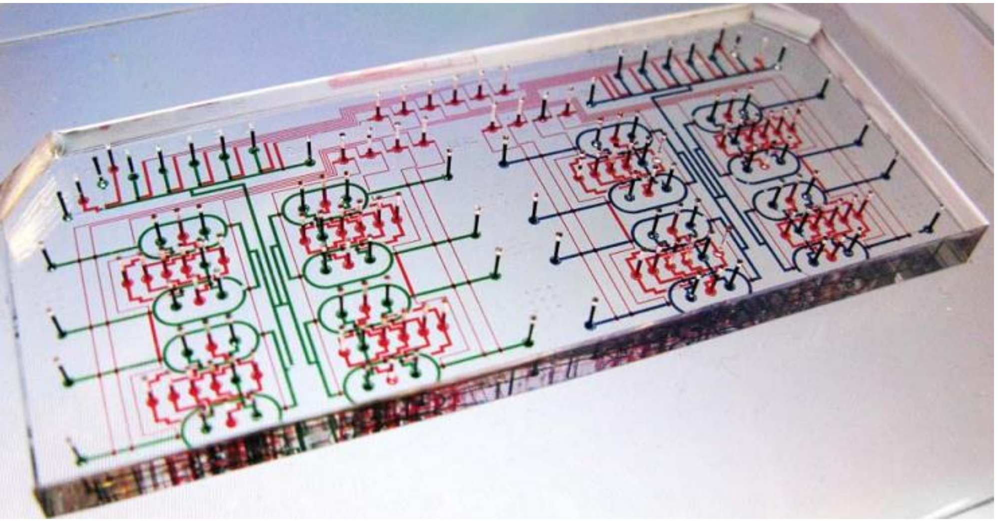
Continuous flow in the flow microchannels

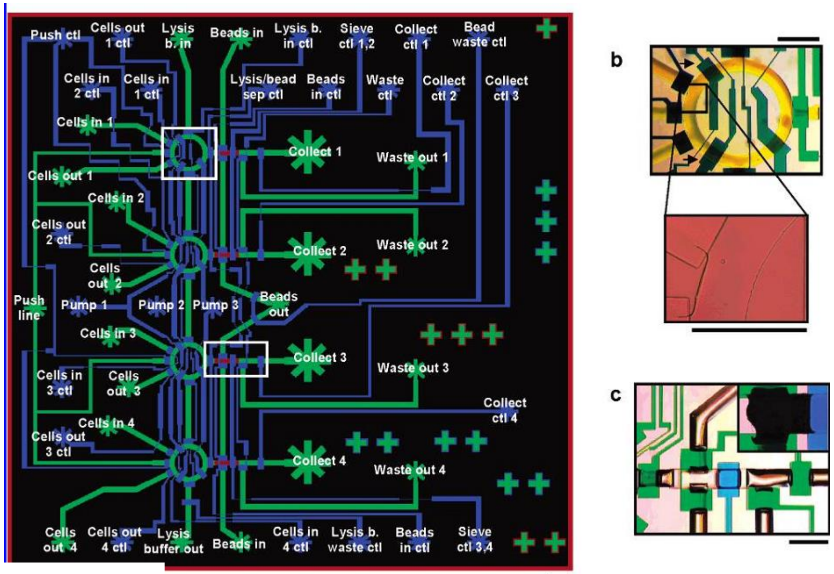




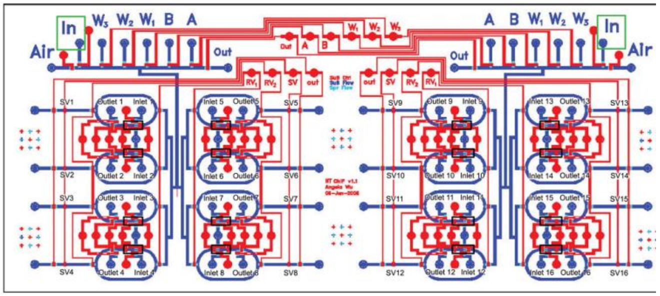
T. Thorsen, S. J. Maerkl, and S. R. Quake, “Microfluidic Large-scale Integration.,” Science, Oct. 2002.

Examples of Flow-Based Microfluidic Biochips (Fluidigm/Stanford, MIT, 2003-2012)

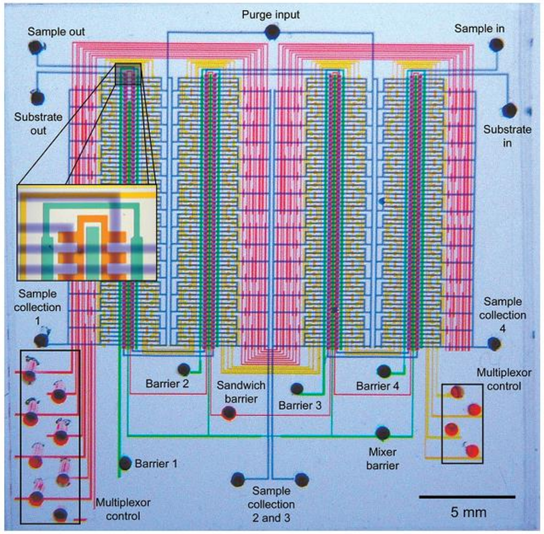


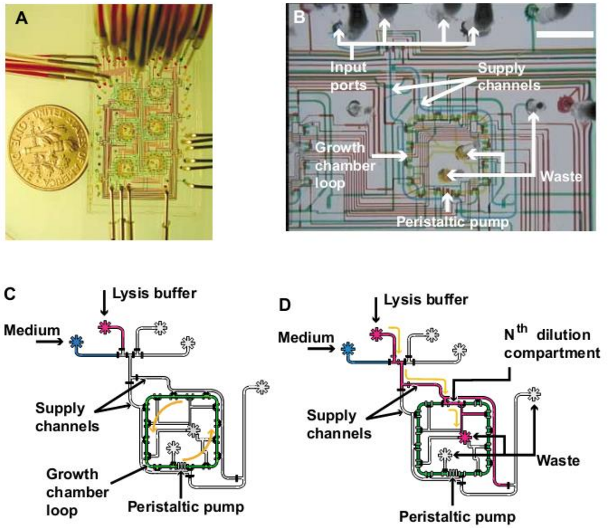


[Wu, Stanford 2012]

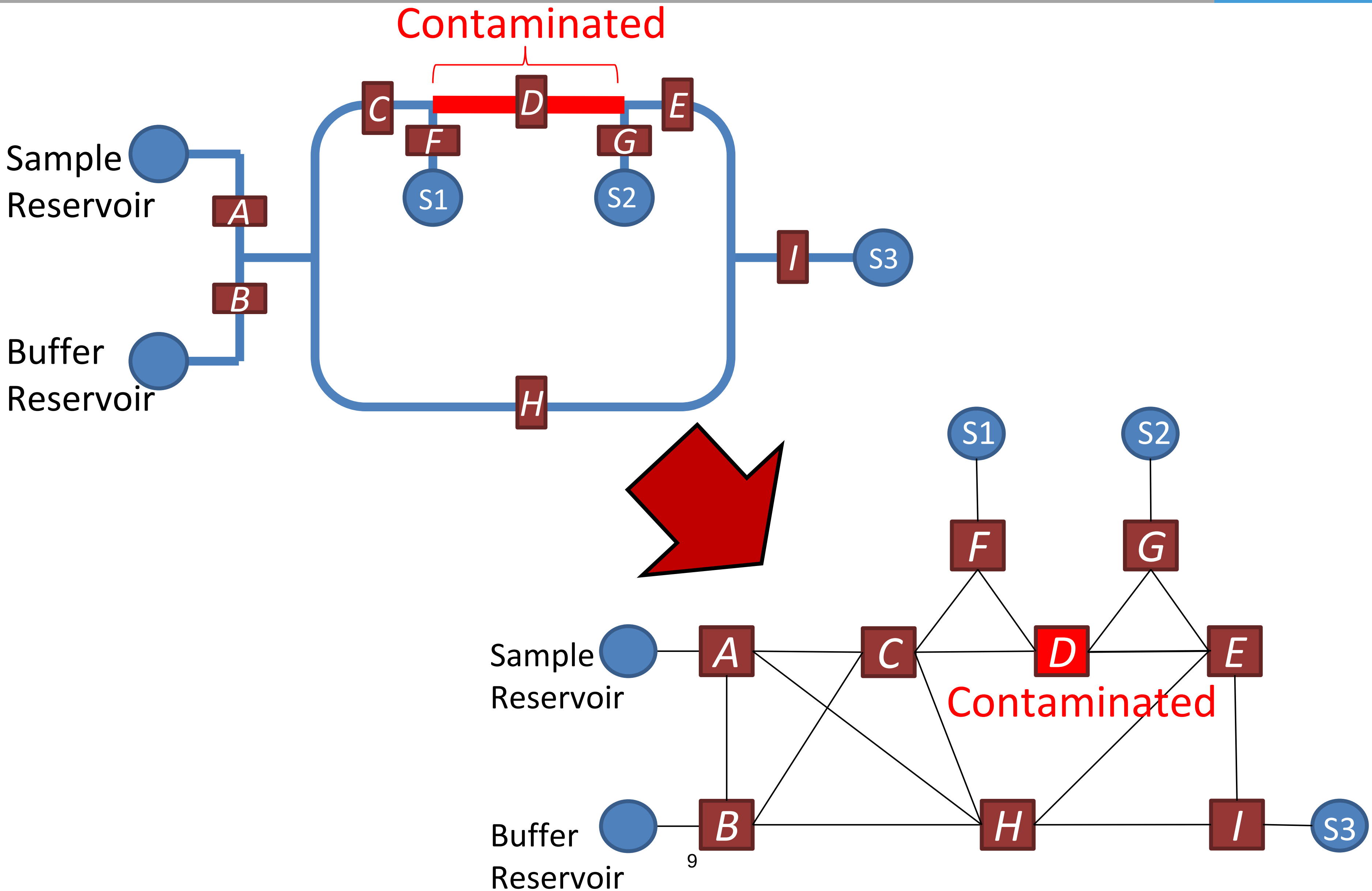


[Wu, Stanford 2012]

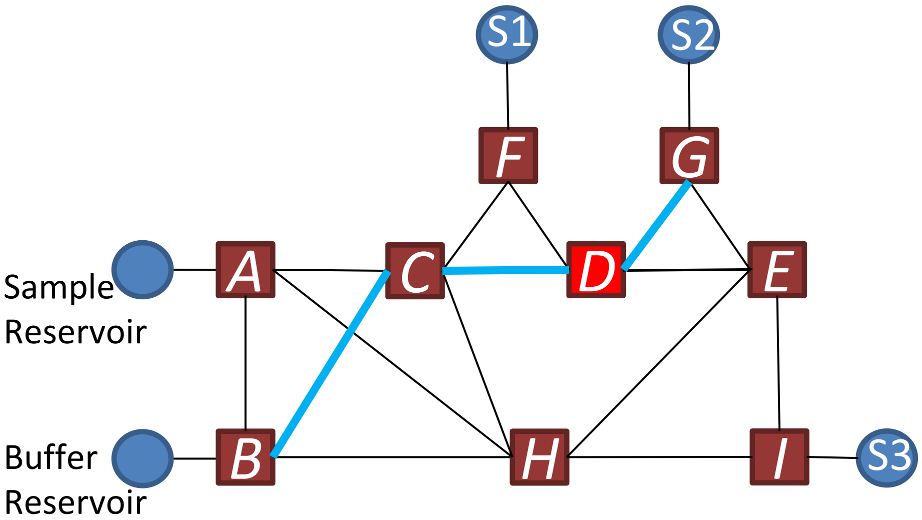




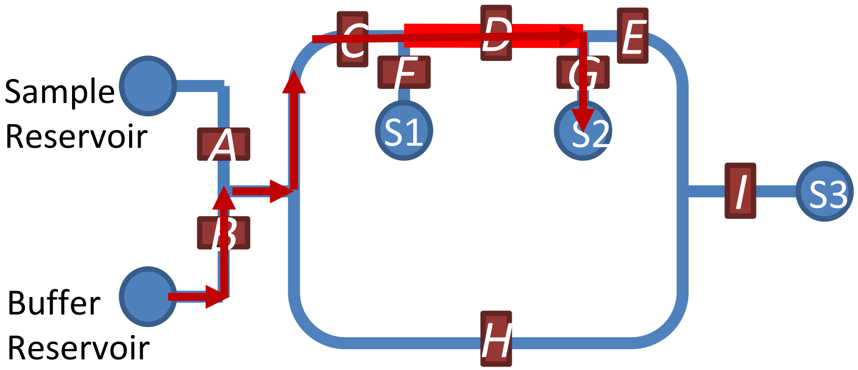
Channel Discretization



Wash Paths



Washing path: $B \rightarrow C \rightarrow D \rightarrow G$

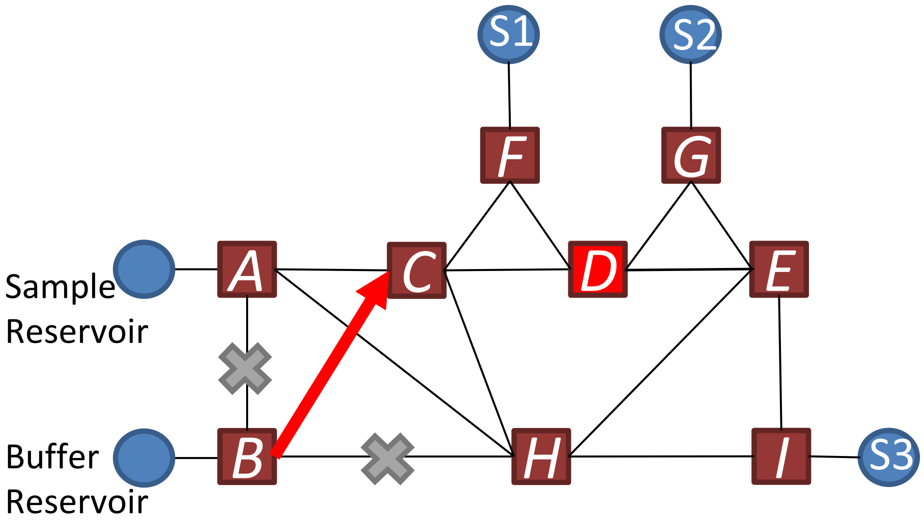


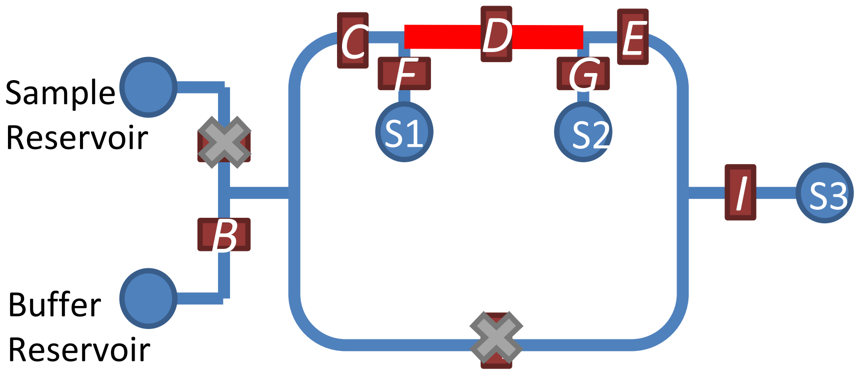
Open valves: {B, C, D, G}

Close valves: {A, F, E, H, I}

Other effective wash paths:
{B, C, D, E, I} & {B, H, E, D, F}

Implementable Paths in a Graph





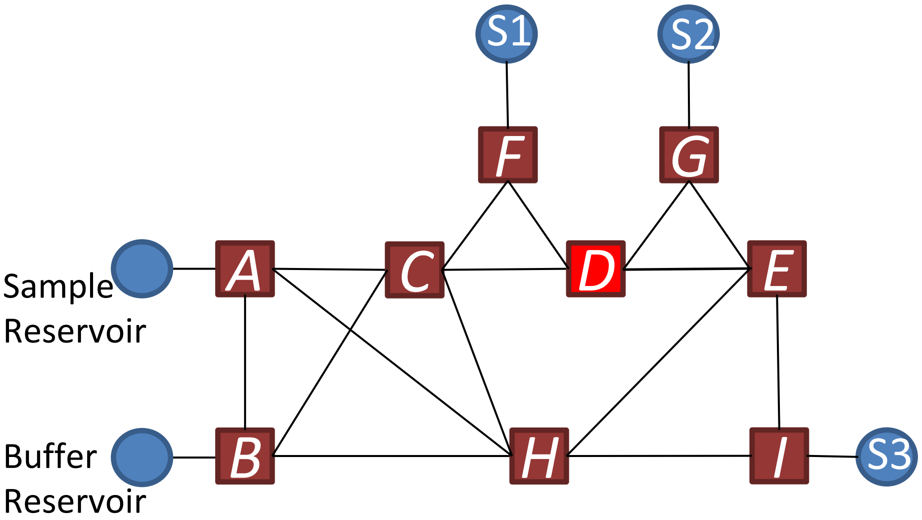
Criteria 1: Start at a buffer reservoir and end at a sink.
Criteria 2: Every vertex can be passed only once.
Criteria 3: At most two valves can be open at each intersection.

Goals and Constraints

- Goal
  - Cover all wash targets
  - Bypass “busy” channels (occupied channels)
  - Wash time minimization
- Difficulties:
  - Paths should be implementable for the given low-degree graph.

Generation of Path Dictionary

- Solutions:
  - Pre-establish a path dictionary to store all implementable paths.
    - Depth-first path search
    - Random path search



Path ID	List of Vertices
1	B, C, F
2	B, C, D, G
3	B, C, D, E, I
4	B, H, I
5	B, H, E, G
6	B, H, E, D, F
7	B, A

Storage of Path Dictionary

- Number of paths >> Number of wash targets
- Only paths that contain a wash targets are candidates washing path.

Path-by-Path

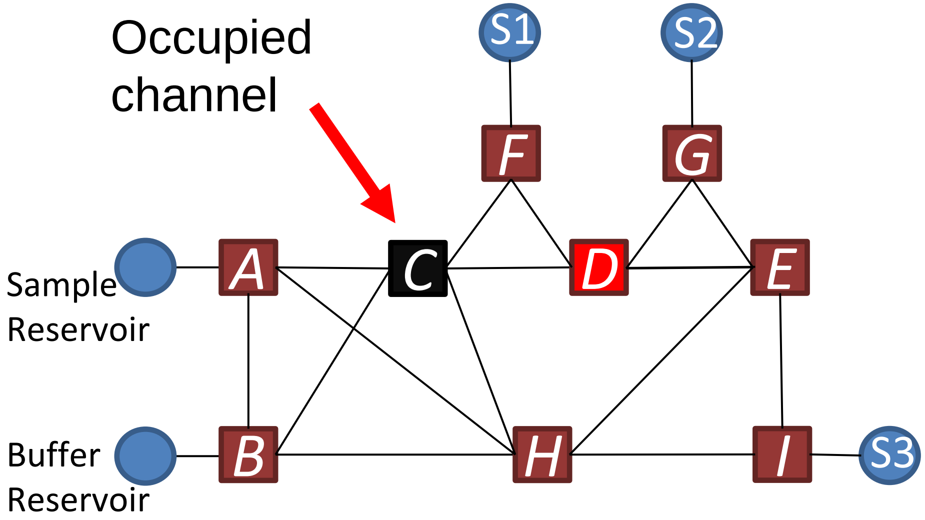
Path ID	List of Vertices
1	B, C, F
2	B, C, D, G
3	B, C, D, E, I
4	B, H, I
5	B, H, E, G
6	B, H, E, D, F
7	B, A

→

Vertex-by-Vertex

Vertex ID	Path List
A	7
B	1, 2, 3, 4, 5, 6
C	1, 2, 3
D	2, 3, 6
E	3, 5, 6
F	1, 6
G	2, 5
H	4, 5, 6
I	3, 4

Vertices Search in Path Dictionary



Vertex ID	Path List
B	1, 2, 3, 4, 5, 6
C	1, 2, 3
D	~~2, 3~~, 6
E	3, 5, 6
F	1, 6
G	2, 5
H	4, 5, 6
I	3, 4

Compressed Path Dictionary

Vertex ID	Path List
D	6

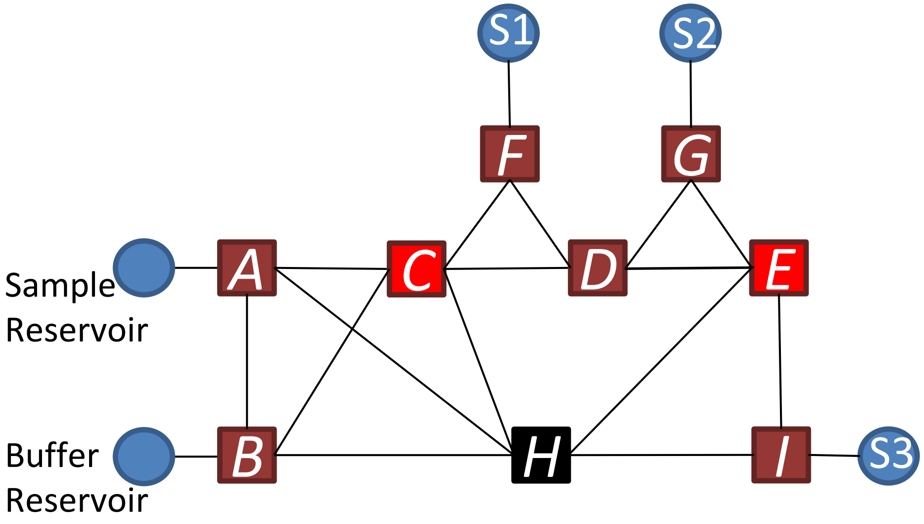
Small dictionary size without any useless data

Advantages:

1) low memory, 2) quick search.

Examples

Wash targets: $\mathcal{T} = \{C, E\}$,
and occupied channel: $\mathcal{O} = \{H\}$.



Vertex ID	Path List
B	1, 2, 3, 4, 5, 6
C	1, 2, 3
D	2, 3, 6
E	3, ~~5, 6~~
F	1, 6
G	2, 5
H	4, 5, 6
I	3, 4

Compressed Dictionary

Vertex ID	Path List
C	1, 2, 3
E	3

Goal: to find a path set, H, that covers all subsets in the compressed dictionary , $\mathbb{S}'$.

Weighted Hitting-set Problems

A hitting set H of a collection $\mathbb{S}'$: a set that contains at least one element from each subset $S \in \mathbb{S}'$, that is, $S \cap H \neq \emptyset, \forall S \in \mathbb{S}'$.

$w(p)$: wash time of a path p

A weighted hitting–set problem:

$$minimize: \sum_{p \in H} w(p)$$ ← Wash time minimization

$$subject\ to: S \cap H \neq \emptyset, \forall S \in \mathbb{S}'$$ ← Cover all wash targets

Goal: to find a path set, H, that covers all subsets in the compressed dictionary, $\mathbb{S}'$.

Compressed Dictionary , $\mathbb{S}'$.

Vertex ID	Path List
C	1, 2, 3
E	3

Wash-optimization Problems

A variant of the weighted hitting-set problem:

$$minimize: \sum_{p \in H} w(p)$$ Wash time minimization

$$subject\ to: S_j \cap H \neq \emptyset, \text{for every } j \in \mathcal{T}$$ Cover all wash targets

$$S_i \cap H = \emptyset, \text{for every } i \in \mathcal{O}.$$ Bypass occupied channels

Symbol	Interpretation in Wash-Optimization Problem
$\mathcal{T}$	Set of wash targets
$\mathcal{O}$	Set of occupied channels
$\mathbb{S}$	Path dictionary
$\mathbb{S}'$	Compressed path dictionary
i, j	A vertex in the graph
S_i, S_j	Set of paths that covers the j^{th} vertex
p	A path in the graph
$w(p)$	Wash time of path p
H	A wash path set

$w(p) = t + l/v$:

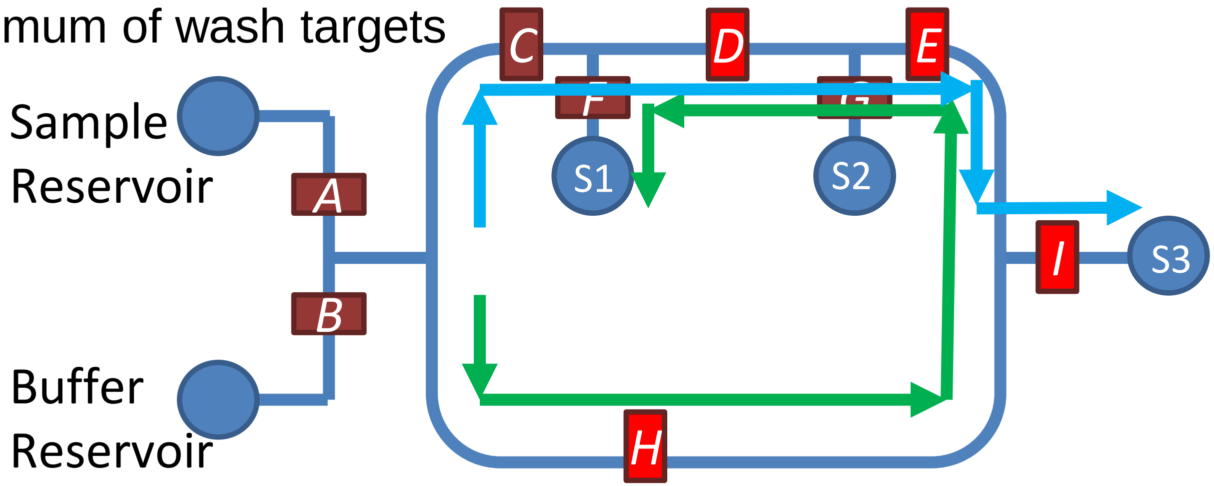
t: setup time, a constant for all paths

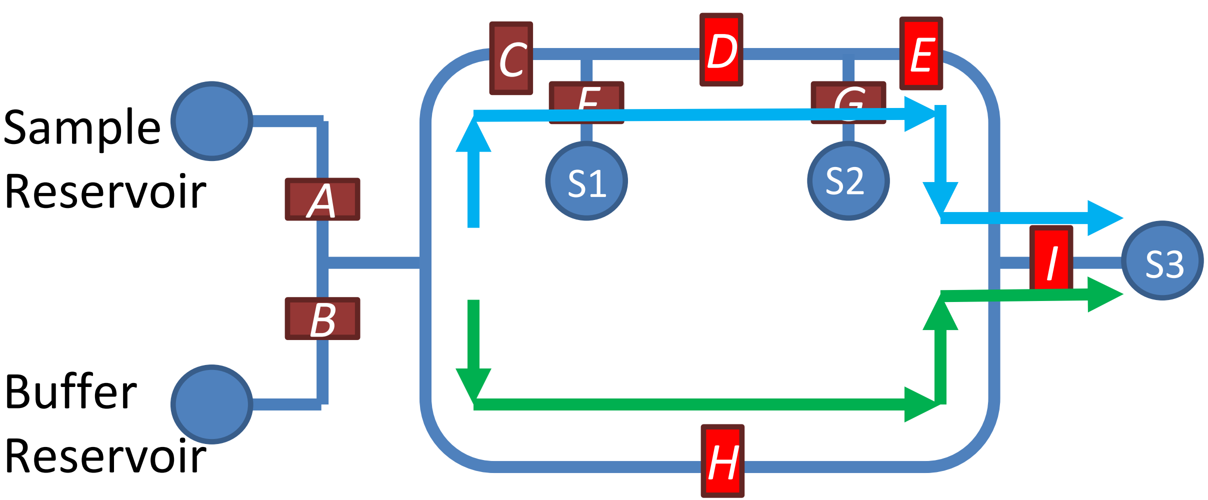
l/v: time for buffer to flow through a path, a variable

Approaches for Hitting-set Problems

Standard algorithms for Hitting-set Problems: a greedy approach that selects in each iteration the paths that contains:

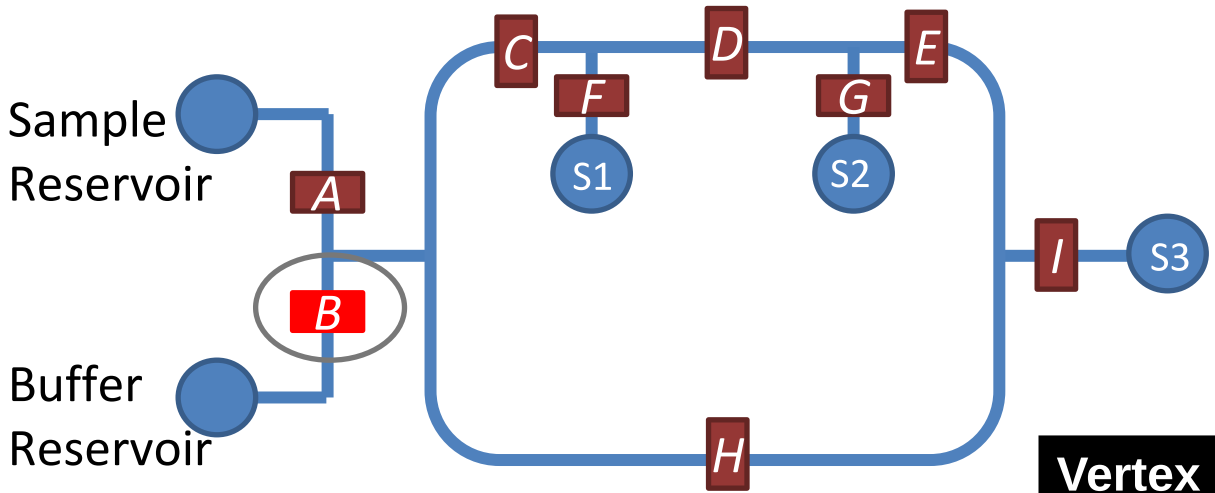
i) the longest paths
ii) the paths contains a maximum of wash targets





Best Wash Paths

Wash Priority of a Vertex: I

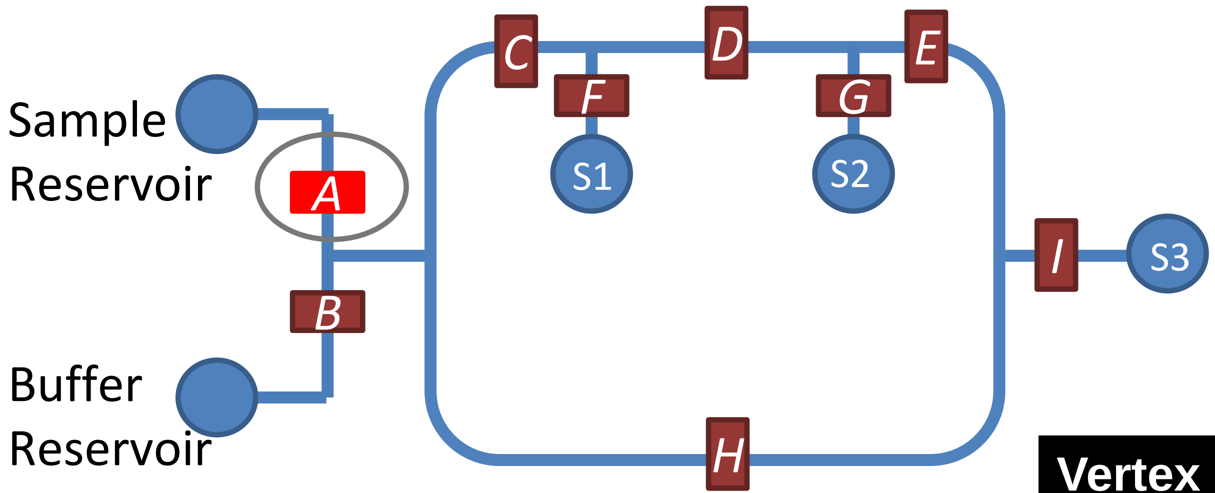


Vertex ID	Path List
A	7
B	1, 2, 3, 4, 5, 6, 7
C	1, 2, 3
D	2, 3, 6
E	3, 5, 6
F	1, 6
G	2, 5
H	4, 5, 6
I	3, 4

Channel B is covered by all paths
-> Channel B need not be considered.

$|M'|$: the number of paths containing wash targets

Wash Priority of a Vertex: II



Vertex ID	Path List
A	7
B	1, 2, 3, 4, 5, 6, 7
C	1, 2, 3
D	2, 3, 6
E	3, 5, 6
F	1, 6
G	2, 5
H	4, 5, 6
I	3, 4

Channel A is covered by Path 7 only.
-> Path 7 must be included in H if Vertex A needs to be washed.

Conclusion: Vertices covered by more paths need to be given lower priorities because they can be targeted more easier at subsequent steps.

Utility Function

A utility function to evaluate "wash priority" of a channel/vertex.

$$f(j) = \log_a \frac{|S'_j| - 1}{|M'| - 1}$$

a : the index to adjust nonlinearity
$|S'_j|$: the number of paths that covers the j^{th} vertex
$|M'|$: the number of paths containing wash targets

Vertex ID	Path List	Utility (a=0.5)
A	7	∞
B	1, 2, 3, 4, 5, 6, 7	0
C	1, 2, 3	1.58
D	2, 3, 6	1.58
E	3, 5, 6	1.58
F	1, 6	2.59
G	2, 5	2.59
H	4, 5, 6	1.59
I	3, 4	2.59

Vertex A will be washed at once.

Vertex B need not be considered at all.

Vertex covered by fewer paths are less likely to be washed.

Wash Value of a Path

Wash value of a path p, $V(p)$, is the sum of utilities of all the covered wash targets.

$$V(p) = \sum f(j)\,, j \in \{j; p \in S_j\}$$

To maximize efficiency, the path with highest value-to-cost ratio, $V(p)/\mathrm{w(p)}$, is selected.

Algorithm 1 The enhanced wash-optimization approach

1: Remove S_j in $\mathbb{S}$, if $j \notin \mathcal{T}$;
2: Remove all paths that include forbidden vertices from the $\mathbb{S}$;
3: Initialization H: $H = \emptyset$;
4: **while** $\mathbb{S} \neq \emptyset$ **do**
5: Update $f(j)$: $f(j) = \log_a \frac{|S_j|-1}{|M|-1}$ for every vertex j, where $M = \cup_{j \in X} S_j$;
6: Update $V(p)$: $V(p) = \sum f(j)$, $j \in \{j : p \in \mathcal{S}_j\}$;
7: Find the path e with maximum value-to-cost ratio, $V(p)/w(p)$;
8: $\mathbb{S} = \mathbb{S} - \mathcal{S}_j$ for every $j \in \{j : e \in S_j\}$;
9: $H = H \cup e$;
10: **end while**
11: Return H

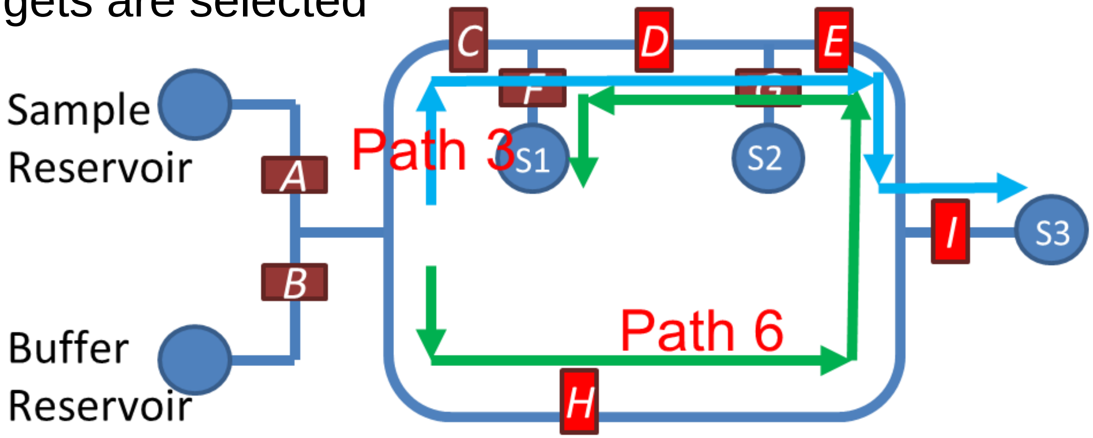
5. Update Utilities of each wash targets

6. Update wash value of each paths

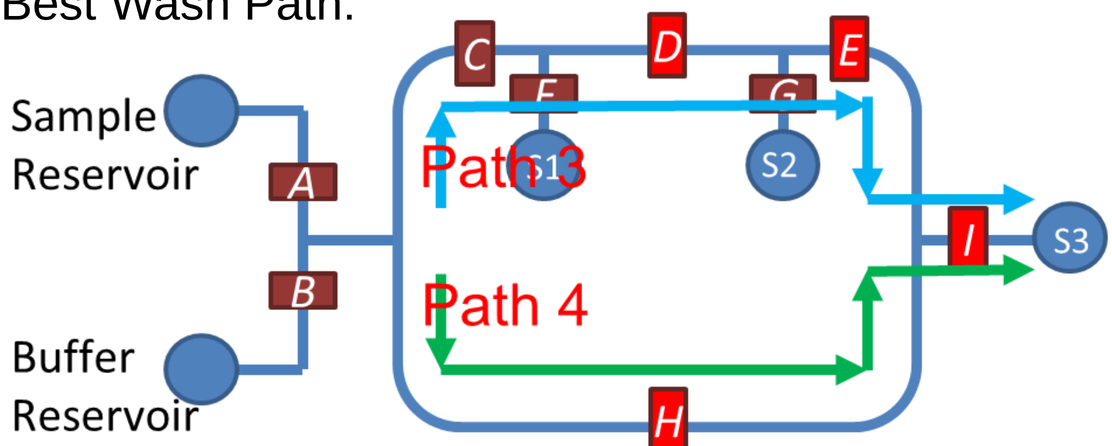
7. The path with highest wash efficiency is selected in each iteration.

Wash Targets $\mathcal{T} = \{\mathrm{D, E, H, I}\}$

The paths contains a maximum of wash targets are selected



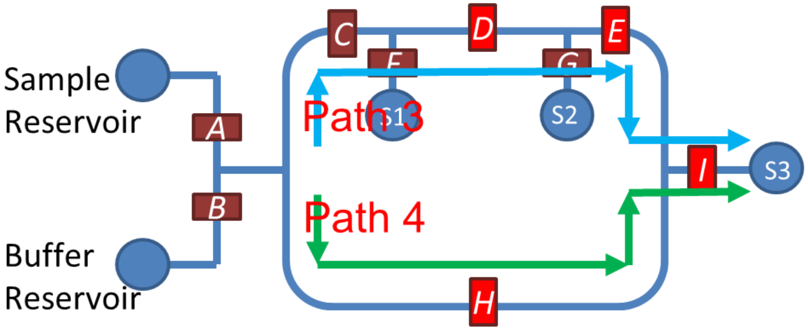
Best Wash Path:



Path Dictionary:

Path ID	List of Vertices
1	B, C, F
2	B, C, D, G
3	B, C, D, E, I
4	B, H, I
5	B, H, E, G
6	B, H, E, D, F
7	B, A

Wash Path Selection Based on Wash Value Evaluation



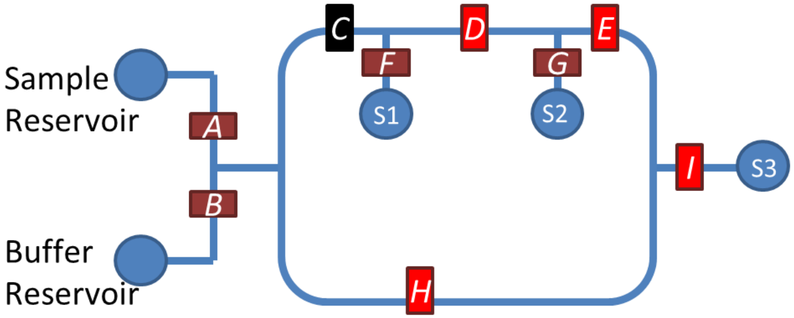
Path Dictionary:

Vertex ID	Path List
~~A~~	~~7~~
~~B~~	~~1, 2, 3, 4, 5, 6, 7~~
~~C~~	~~1, 2, 3~~
D	2, 3, 6
E	3, 5, 6
~~F~~	~~1, 6~~
~~G~~	~~2, 5~~
H	4, 5, 6
I	3, 4

Vertex ID	Path List	Utilities
D	2, 3, 6	1
E	3, 5, 6	1
H	4, 5, 6	1
I	3, 4	2

Path ID	List of Vertices	$V(p)$	$w(p)$	$V(p)/w(p)$
2	B, C, D, G	1	5	0.2
3	B, C, D, E, I	4	5	0.8
4	B, H, I	3	5	0.6
5	B, H, E, G	2	6	0.3
6	B, H, E, D, F	3	7	0.4

If $\mathcal{O} = \{C\}, \mathcal{T} = \{\mathrm{D, E, H, I}\}, \ldots$



Path Dictionary:

Vertex ID	Path List
~~A~~	~~7~~
~~B~~	~~1, 2, 3, 4, 5, 6, 7~~
C	1, 2, 3
D	2, 3, 6
E	3, 5, 6
~~F~~	~~1, 6~~
~~G~~	~~2, 5~~
H	4, 5, 6
I	3, 4

Vertex ID	Path List	Utilities
D	6	∞
E	5, 6	1
H	4, 5, 6	0
I	4	∞

Path ID	List of Vertices	$V(p)$	$w(p)$	$V(p)/\mathrm{w(p)}$
4	B, H, I	∞	5	∞
5	B, H, E, G	1	6	0.3
6	B, H, E, D, F	∞	7	∞

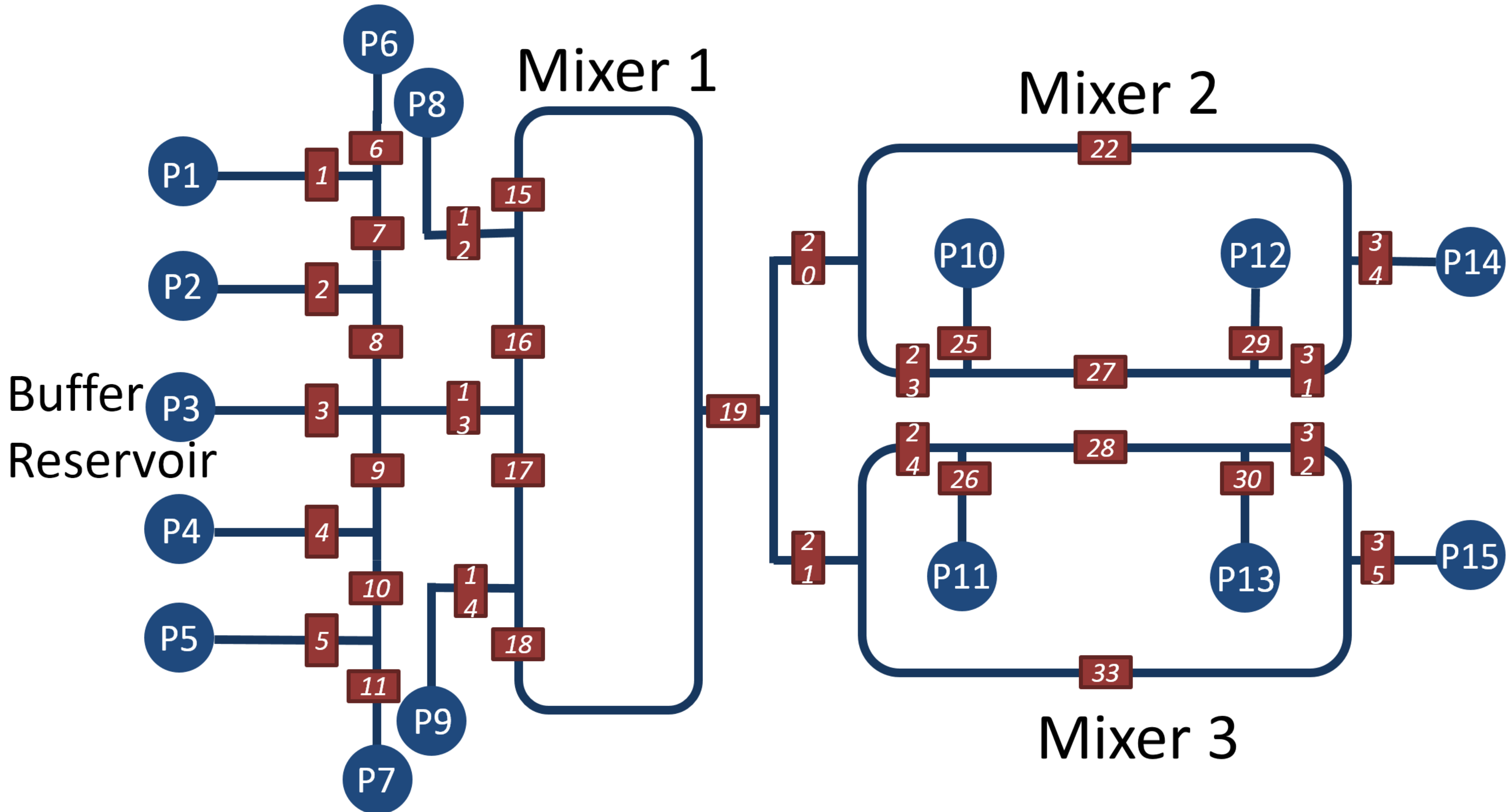
Algorithm Evaluation

- Method I: A path dictionary is not generated. Buffer flows along the "longest" path to cover as many microchannels as possible.

- Method II: The paths that contains a maximum of washing targets in the dictionary is selected.

- Method III: Wash priorities are considered and the most “efficient” wash path is selected.

Application to Fabricated Biochip: I



The path dictionary contains 34 paths.

A. Wu et al., "Automated Microfluidic Chromatin Immunoprecipitation from 2,000 Cells," Lab on a Chip, vol. 9, 2009.

Path Dictionary

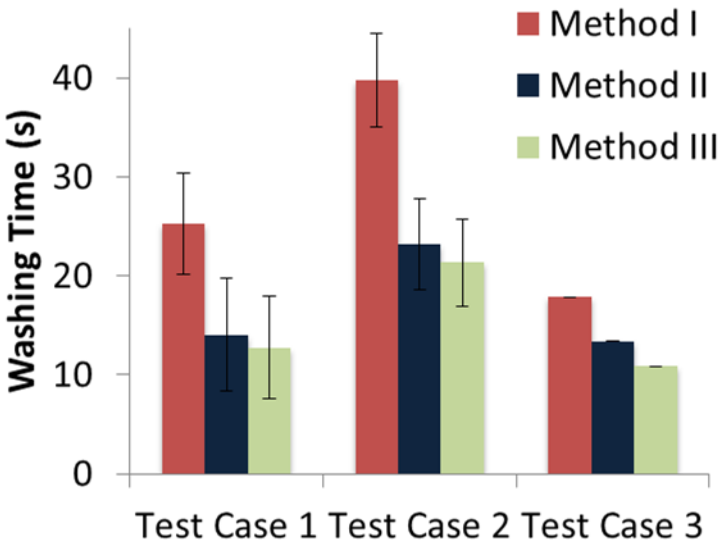
Entry #	Microchannels	Entry #	Microchannels
1	3,13,17,14	18	3,13,16,15,19,20,23,27,29
2	3,13,17,18,15,12	19	3,13,16,15,19,20,23,27,31,34
3	3,13,17,18,19,20,23,25	20	3,13,16,15,19,20,22,31,27,25
4	3,13,17,18,19,20,23,27,29	21	3,13,16,15,19,20,22,31,29
5	3,13,17,18,19,20,23,27,31,34	22	3,13,16,15,19,20,22,34
6	3,13,17,18,19,20,22,31,27,25	23	3,13,16,15,19,21,24,26
7	3,13,17,18,19,20,22,31,29	24	3,13,16,15,19,21,24,28,30
8	3,13,17,18,19,20,22,34	25	3,13,16,15,19,21,24,28,32,35
9	3,13,17,18,19,21,24,26	26	3,13,16,15,19,21,33,32,28,26
10	3,13,17,18,19,21,24,28,30	27	3,13,16,15,19,21,33,32,30
11	3,13,17,18,19,21,24,28,32,35	28	3,13,16,15,19,21,33,35
12	3,13,17,18,19,21,33,32,28,26	29	3,8,2
13	3,13,17,18,19,21,33,32,30	30	3,8,7,1
14	3,13,17,18,19,21,33,35	31	3,8,7,6
15	3,13,16,12	32	3,9,4
16	3,13,16,15,18,14	33	3,9,10,5
17	3,13,16,15,19,20,23,25	34	3,9,10,11

Additional Information about Path Dictionary

Micro-channel ID	Length (mm)	# of Covering Paths	Utility Function Value	Micro-channel ID	Length (mm)	# of Covering Paths	Utility Function Value
1	2.5	1	inf	19	1	24	0.52
2	2.5	1	inf	20	3.5	12	1.59
3	2.5	34	0	21	3.5	12	1.59
4	2.5	1	inf	22	11	6	2.72
5	2.5	1	inf	23	3.65	6	2.72
6	1	1	inf	24	3.65	6	2.72
7	1	2	5.04	25	1.5	4	3.46
8	1	3	4.04	26	1.5	4	3.46
9	1	3	4.04	27	4	6	2.72
10	1	2	5.04	28	4	6	2.72
11	1	1	inf	29	1.5	4	3.46
12	2.5	2	5.04	30	1.5	4	3.46
13	3.5	28	0.29	31	3.65	6	2.72
14	2.5	2	5.04	32	3.65	6	2.72
15	9.3	14	1.34	33	11	6	2.72
16	2	14	1.34	34	1.5	4	3.46
17	2	14	1.34	35	1.5	4	3.46
18	9.3	14	1.34				

Results I

Wash time:



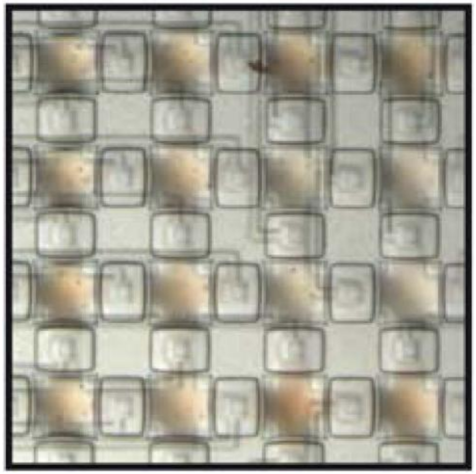
CPU time:

Test Case	Method I (ms)	Method II (ms)	Method III (ms)
1	15.95	2.78	7.61
2	25.75	5.48	20.72
3	10.23	1.87	3.41

Test Case 1: 10 wash targets and 4 occupied microchannels
Test Case 2: 20 wash targets and 6 occupied microchannels
Test Case 3: 13 wash targets and 1 occupied microchannels

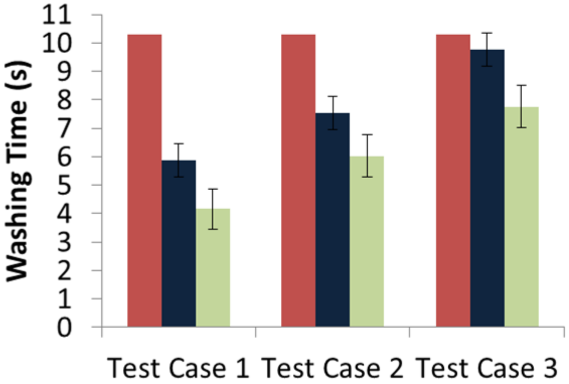
Application to Fabricated Biochip: II

A programmable microfluidic device with an 8-by-8 grid*



Test Case 1: 20 wash targets and 8 occupied microchannels
Test Case 2: 40 wash targets and 10 occupied microchannels
Test Case 3: 80 wash targets and 15 occupied microchannels

Wash time:



CPU time:

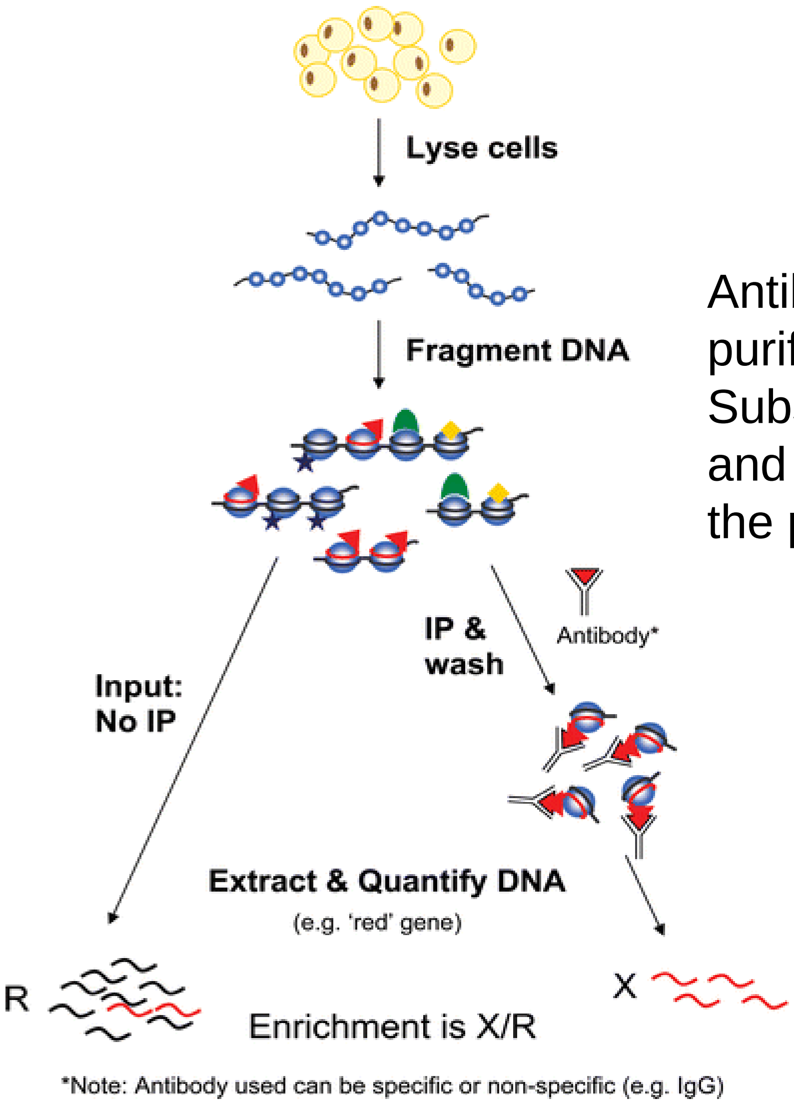
Test Case	Method I (ms)	Method II (ms)	Method III (ms)
1	30.8	0.35	2.38
2	45.2	0.42	4.32
3	69.5	0.66	6.20

* L. M. Fidalgo and S. J. Maerkl, "A software-programmable microfluidic device for automated biology.," Lab on a chip, vol. 11, pp. 1612–9, 2011

Conclusions

- The first approach for automated wash optimization in flow-based microfluidic biochips
- Wash optimization problem is formulated as a variant of hitting-set problem.
- A utility function to evaluate the wash priorities of washing targets
- Occupied channel is bypassed to avoid interruption of other concurrent fluid-handling tasks.

Chromatin immunoprecipitation (CHiP)



A schematic of the ChIP process flow.

Antibodies against the proteins of interest are used to purify these proteins along with the DNA they bind to. Subsequently this DNA can be released, identified and quantified, giving information about where the protein binds across the genome.