On the Space Complexity of Set Agreement

Carole Delporte-Gallet	*LIAFA, Université Paris-Diderot*	*France*
Hugues Fauconnier	*LIAFA, Université Paris-Diderot*	*France*
Petr Kuznetsov	*Télécom ParisTech*	*France*
Eric Ruppert	*York University*	*Canada*

July 22, 2015

Agreement Using Registers

Solving agreement using read/write registers.

		wait-free
consensus		

Agreement Using Registers

Solving agreement using read/write registers.

		wait-free
consensus		✗

Agreement Using Registers

Solving agreement using read/write registers.

	obstruction-free	wait-free
consensus	✔	✘

Agreement Using Registers

Solving agreement using read/write registers.

	obstruction-free	wait-free
consensus	✔	✘
set-agreement	✔	

Agreement Using Registers

Solving agreement using read/write registers.

	obstruction-free	wait-free
consensus	✔	✘
set-agreement	✔	✘

Parameters

m-obstruction-free

obstruction-free				m			wait-free
	1	2	3		$n-2$	$n-1$	n

An algorithm is *m-obstruction-free* if some process is guaranteed to terminate when at most m processes continue to take steps.

Parameters

m-obstruction-free *k*-set agreement

		obstruction-free			*m*			wait-free
		1	2	3		$n-2$	$n-1$	n
consensus	1							
	2							
	3							
k	4							
	$n-2$							
set-agreement	$n-1$							

k-set agreement: processes must output at most *k* different values.

Parameters

m-obstruction-free k-set agreement for n processes

		obstruction-free			m			wait-free
		1	2	3		$n-2$	$n-1$	n
consensus	1				$\cdots$			
	2				$\cdots$			
	3				$\cdots$			
k	4				$\cdots$			
		$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
	$n-2$				$\cdots$			
set-agreement	$n-1$				$\cdots$			

Parameters

m-obstruction-free *k*-set agreement for *n* processes

		obstruction-free			*m*			wait-free
		1	2	3		$n-2$	$n-1$	n
consensus	1				⋯			✘
	2				⋯			✘
	3				⋯			✘
k	4				⋯			✘
		⋮	⋮	⋮	⋱	⋮	⋮	⋮
	$n-2$				⋯			✘
set-agreement	$n-1$				⋯			✘

Parameters

m-obstruction-free *k*-set agreement for *n* processes

		obstruction-free			*m*			wait-free
		1	2	3		$n-2$	$n-1$	n
consensus	1	✔			⋯			✘
	2	✔			⋯			✘
	3	✔			⋯			✘
k	4	✔			⋯			✘
		⋮	⋮	⋮	⋱	⋮	⋮	⋮
	$n-2$	✔			⋯			✘
set-agreement	$n-1$	✔			⋯			✘

Parameters

m-obstruction-free k-set agreement for n processes

		obstruction-free			m			wait-free
		1	2	3		$n-2$	$n-1$	n
consensus	1	✔	✘	✘	⋯	✘	✘	✘
	2	✔	✔	✘	⋯	✘	✘	✘
	3	✔	✔	✔	⋯	✘	✘	✘
k	4	✔	✔	✔	⋯	✘	✘	✘
		⋮	⋮	⋮	⋱	⋮	⋮	⋮
	$n-2$	✔	✔	✔	⋯	✔	✘	✘
set-agreement	$n-1$	✔	✔	✔	⋯	✔	✔	✘

Space Complexity: Known Results

Problem: m-obstruction-free k-set agreement for n processes ($m \leq k < n$)

How many registers are needed?

Previous work

- n (single-writer) registers are sufficient
- For $m = k = 1$, $\Omega(\sqrt{n})$ registers needed [FHS98]
- For $m = 1$, $2n - 2k$ registers are sufficient [DFGR13]

Space Complexity: Known Results

Problem: m-obstruction-free k-set agreement for n processes ($m \leq k < n$)

How many registers are needed?

Previous work

- n (single-writer) registers are sufficient
- For $m = k = 1$, $\Omega(\sqrt{n})$ registers needed [FHS98]
- For $m = 1$, $2n - 2k$ registers are sufficient [DFGR13]

Repeated Agreement

Repeated k-set agreement problem

- Series $A_1, A_2, A_3, \ldots$ of set agreement instances
- In each instance A_i, processes output at most k different values
- Processes access instances in order

Motivation

- Herlihy's universal construction (with $k = 1$).
- Possible route to lower bound for one-shot problem.

Repeated Agreement

Repeated k-set agreement problem

- Series $A_1, A_2, A_3, \ldots$ of set agreement instances
- In each instance A_i, processes output at most k different values
- Processes access instances in order

Motivation

- Herlihy's universal construction (with $k = 1$).
- Possible route to lower bound for one-shot problem.

Our Results

Bounds on number of registers needed for
m-obstruction-free k-set agreement for n processes

	Repeated	One-Shot
Non-Anon.	$\geq n+m-k$ $\leq n+2m-k$ $\leq n+m-k$ (known ids)	≥ 2 [DFGR13] $\leq n+2m-k$ $\leq n+m-k$ (known ids)
Anon.	$\geq n+m-k$ $\leq (m+1)(n-k)+m^2+1$	$\geq \sqrt{m(\frac{n}{k}-2)}$ $\leq (m+1)(n-k)+m^2$

- Bounds show dependence on k and m
- First anonymous set agreement algorithm
- $\Omega(\sqrt{n})$ lower bound when $m=k=1$ is a special case
- Bounds are nearly tight for repeated (non-anonymous)

YORK UNIVERSITÉ UNIVERSITY

Our Results

Bounds on number of registers needed for
m-obstruction-free k-set agreement for n processes

	Repeated	One-Shot
Non-Anon.	$\geq n+m-k$ $\leq n+2m-k$ $\leq n+m-k$ (known ids)	≥ 2 [DFGR13] $\leq n+2m-k$ $\leq n+m-k$ (known ids)
Anon.	$\geq n+m-k$ $\leq (m+1)(n-k)+m^2+1$	$\geq \sqrt{m(\frac{n}{k}-2)}$ $\leq (m+1)(n-k)+m^2$

- Bounds show dependence on k and m
- First anonymous set agreement algorithm
- $\Omega(\sqrt{n})$ lower bound when $m=k=1$ is a special case
- Bounds are nearly tight for repeated (non-anonymous)

Our Results

Bounds on number of registers needed for
m-obstruction-free k-set agreement for n processes

	Repeated	One-Shot
Non-Anon.	$\geq n+m-k$ $\leq n+2m-k$ $\leq n+m-k$ (known ids)	≥ 2 [DFGR13] $\leq n+2m-k$ $\leq n+m-k$ (known ids)
Anon.	$\geq n+m-k$ $\leq (m+1)(n-k)+m^2+1$	$\geq \sqrt{m(\frac{n}{k}-2)}$ $\leq (m+1)(n-k)+m^2$

- Bounds show dependence on k and m
- First anonymous set agreement algorithm
- $\Omega(\sqrt{n})$ lower bound when $m=k=1$ is a special case
- Bounds are nearly tight for repeated (non-anonymous)

Repeated Set Agreement Bounds

Repeated *m*-obstruction-free *k*-set agreement for *n* processes

		obstruction-free			m			wait-free
		1	2	3		$n-2$	$n-1$	n
consensus	1	✔	✘	✘	⋯	✘	✘	✘
	2	✔	✔	✘	⋯	✘	✘	✘
	3	✔	✔	✔	⋯	✘	✘	✘
k	4	✔	✔	✔	⋯	✘	✘	✘
		⋮	⋮	⋮	⋱	⋮	⋮	⋮
	$n-2$	✔	✔	✔	⋯	✔	✘	✘
set-agreement	$n-1$	✔	✔	✔	⋯	✔	✔	✘

Repeated Set Agreement Bounds

Repeated m-obstruction-free k-set agreement for n processes

obstruction-free — m — wait-free

k \ m	1	2	3		$n-2$	$n-1$	n
consensus 1	$\geq n$ $\leq n+1$	✘	✘	⋯	✘	✘	✘
2	$\geq n-1$ $\leq n$	$\geq n$ $\leq n+2$	✘	⋯	✘	✘	✘
3	$\geq n-2$ $\leq n-1$	$\geq n-1$ $\leq n+1$	$\geq n$ $\leq n+3$	⋯	✘	✘	✘
4	$\geq n-3$ $\leq n-2$	$\geq n-2$ $\leq n$	$\geq n-1$ $\leq n+2$	⋯	✘	✘	✘
⋮	⋮	⋮	⋮	⋱	⋮	⋮	⋮
$n-2$	≥ 3 ≤ 4	≥ 4 ≤ 6	≥ 5 ≤ 8	⋯	$\geq n$ $\leq 2n-2$	✘	✘
set-agreement $n-1$	≥ 2 ≤ 3	≥ 3 ≤ 5	≥ 4 ≤ 7	⋯	$\geq n-1$ $\leq 2n-3$	$\geq n$ $\leq 2n-1$	✘

Repeated Set Agreement Bounds

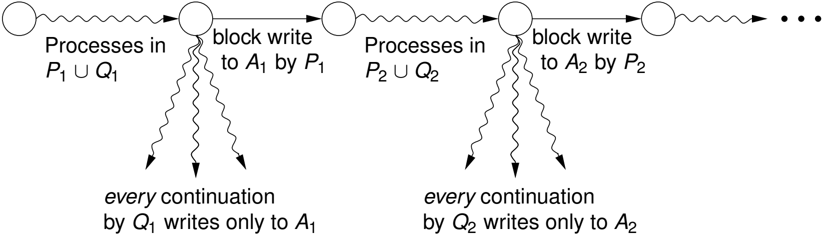
Repeated *m*-obstruction-free *k*-set agreement for *n* processes

obstruction-free — *m* — wait-free

k	1	2	3		$n-2$	$n-1$	n
consensus 1	$\geq n$ $\leq n$	✘	✘	$\cdots$	✘	✘	✘
2	$\geq n-1$ $\leq n$	$\geq n$ $\leq n$	✘	$\cdots$	✘	✘	✘
3	$\geq n-2$ $\leq n-1$	$\geq n-1$ $\leq n$	$\geq n$ $\leq n$	$\cdots$	✘	✘	✘
4	$\geq n-3$ $\leq n-2$	$\geq n-2$ $\leq n$	$\geq n-1$ $\leq n$	$\cdots$	✘	✘	✘
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
$n-2$	≥ 3 ≤ 4	≥ 4 ≤ 6	≥ 5 ≤ 8	$\cdots$	$\geq n$ $\leq n$	✘	✘
set-agreement $n-1$	≥ 2 ≤ 3	≥ 3 ≤ 5	≥ 4 ≤ 7	$\cdots$	$\geq n-1$ $\leq n$	$\geq n$ $\leq n$	✘

Lower Bound for Repeated Set Agreement

Consider any m-obstruction-free k-set agreement algorithm.
Construct an execution:



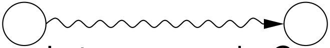
Q_i's are disjoint sets of m processes each.
P_i is set of processes disjoint from $Q_1, Q_2, \ldots, Q_i$.
A_i is a set of registers.

Constructing the Execution

$Q = \{p_1, p_2, p_3, \ldots, p_m\}$
$P = \{\}$
$A = \{\}$

Constructing the Execution

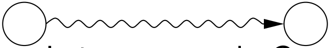
$Q = \{p_1, p_2, p_3, \ldots, p_m\}$
$P = \{\}$
$A = \{\}$



Let processes in Q run until a process p_1 is poised to write to $R_1 \notin A$

Constructing the Execution

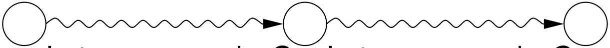
$Q = \{ \quad p_2, p_3, \ldots, p_m, p_{m+1} \}$
$P = \{p_1\}$
$A = \{R_1\}$



Let processes in Q run until a process p_1 is poised to write to $R_1 \notin A$

Constructing the Execution

$Q = \{ \quad p_2, p_3, \ldots, p_m, p_{m+1} \}$
$P = \{p_1\}$
$A = \{R_1\}$

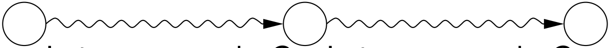


Let processes in Q run until a process p_1 is poised to write to $R_1 \notin A$

Let processes in Q run until a process p_2 is poised to write to $R_2 \notin A$

Constructing the Execution

$Q = \{ \qquad p_3, \ldots, p_m, p_{m+1}, p_{m+2}\}$
$P = \{p_1, p_2\}$
$A = \{R_1, R_2\}$



Let processes in Q run until a process p_1 is poised to write to $R_1 \notin A$

Let processes in Q run until a process p_2 is poised to write to $R_2 \notin A$

Constructing the Execution

$Q = \{ \qquad p_3, \ldots, p_m, p_{m+1}, p_{m+2}\}$
$P = \{p_1, p_2\}$
$A = \{R_1, R_2\}$

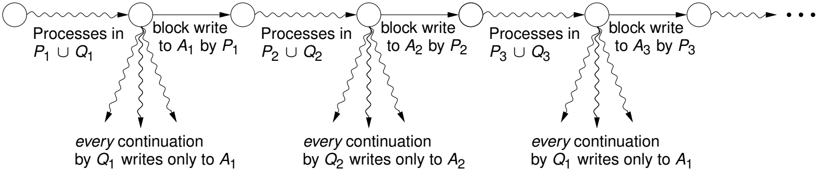
Let processes in Q run until a process p_1 is poised to write to $R_1 \notin A$

Let processes in Q run until a process p_2 is poised to write to $R_2 \notin A$

Repeat until *every* continuation by Q writes only registers in A.

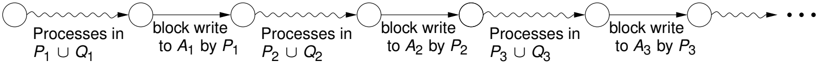
Lower Bound for Repeated Set Agreement

Q_i's are disjoint sets of m processes each.
P_i is set of processes disjoint from $Q_1, Q_2, \ldots, Q_i$.
Let r be the number of registers used by the algorithm.



Lower Bound for Repeated Set Agreement

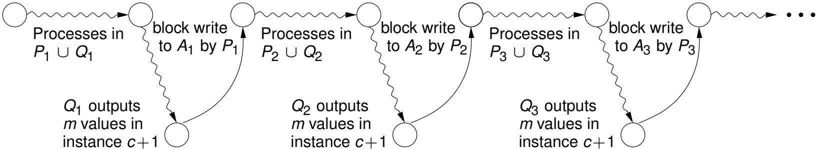
Q_i's are disjoint sets of m processes each.
P_i is set of processes disjoint from $Q_1, Q_2, \ldots, Q_i$.
Let r be the number of registers used by the algorithm.



Let $c = \#$ set agreement instances accessed in this execution.

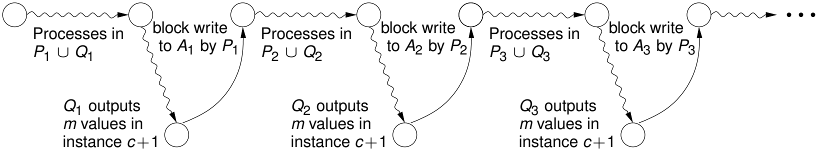
Lower Bound for Repeated Set Agreement

Q_i's are disjoint sets of m processes each.
P_i is set of processes disjoint from $Q_1, Q_2, \ldots, Q_i$.
Let r be the number of registers used by the algorithm.



Lower Bound for Repeated Set Agreement

Q_i's are disjoint sets of m processes each.
P_i is set of processes disjoint from $Q_1, Q_2, \ldots, Q_i$.
Let r be the number of registers used by the algorithm.



$\frac{k+1}{m}$ repetitions yields a contradiction.
This is possible if $n \geq (\frac{k+1}{m} - 1) \cdot m + r = k + 1 - m + r$.
Thus, $n < k + 1 - m + r$.
$\Rightarrow r \geq n + m - k$.

Algorithm for m-Obstruction-Free k-Set Agreement

Use snapshot object A with $n + 2m - k$ components.

Repeat:

1. write $(pref, id)$ into $A[i]$
2. scan A
3. if at most m different pairs, output value from one that appears twice
4. if my pair appears only where I last wrote it AND some other pair (v, id') appears twice then $pref \leftarrow v$
5. else $i++$

Algorithm for m-Obstruction-Free k-Set Agreement

Use snapshot object A with $n + 2m - k$ components.

Repeat:

1. write $(pref, id)$ into $A[i]$
2. scan A
3. if at most m different pairs, output value from one that appears twice
4. if my pair appears only where I last wrote it AND some other pair (v, id') appears twice then $pref \leftarrow v$
5. else $i++$

Example

$n = 5, m = 3, k = 4.$
Use $n + 2m - k = 7$ components.

$v_5, 5$	$v_3, 3$	$v_2, 2$	$v_2, 2$	$v_2, 2$	$v_3, 3$	$v_5, 5$

Example

$n = 5, m = 3, k = 4$.
Use $n + 2m - k = 7$ components.

$v_5, 5$	$v_3, 3$	$v_2, 2$	$v_2, 2$	$v_2, 2$	$v_3, 3$	$v_5, 5$

↑

p_4 (*pref* v_4)

Example

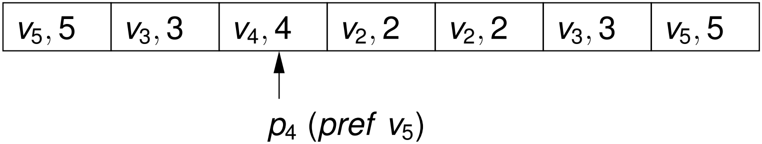
$n = 5, m = 3, k = 4$.
Use $n + 2m - k = 7$ components.

$v_5, 5$	$v_3, 3$	$v_4, 4$	$v_2, 2$	$v_2, 2$	$v_3, 3$	$v_5, 5$

p_4 (*pref* v_4)

Example

$n = 5, m = 3, k = 4.$
Use $n + 2m - k = 7$ components.



Example

$n = 5, m = 3, k = 4.$
Use $n + 2m - k = 7$ components.

$v_5, 5$	$v_3, 3$	$v_5, 4$	$v_2, 2$	$v_2, 2$	$v_3, 3$	$v_5, 5$

↑

p_4 (*pref* v_5)

Example

$n = 5, m = 3, k = 4.$
Use $n + 2m - k = 7$ components.

$v_5, 5$	$v_3, 3$	$v_5, 4$	$v_5, 4$	$v_2, 2$	$v_3, 3$	$v_5, 5$

↑

p_4 (*pref* v_5)

Example

$n = 5, m = 3, k = 4.$
Use $n + 2m - k = 7$ components.

$v_5, 5$	$v_3, 3$	$v_5, 4$	$v_5, 4$	$v_5, 4$	$v_3, 3$	$v_5, 5$

p_4 (*pref* v_5)

Validity

Validity: Every output value is the input of some process.

Trivial proof: values in A are input values of some process.

Agreement

Agreement: At most k different values are output.

We don't care what the first $k - m$ processes output.

Claim: The last $n + m - k$ processes output $\leq m$ values.

When the first of those $n + m - k$ processes does final scan S, it sees at most m different pairs. We prove that afterwards,

- Only pairs with those values can appear in 2 locations.
- No other value can ever be output.

Intuition:

Agreement

Agreement: At most k different values are output.

We don't care what the first $k - m$ processes output.

Claim: The last $n + m - k$ processes output $\leq m$ values.

When the first of those $n + m - k$ processes does final scan S, it sees at most m different pairs. We prove that afterwards,

- Only pairs with those values can appear in 2 locations.
- No other value can ever be output.

Intuition:

Agreement

Agreement: At most k different values are output.

We don't care what the first $k - m$ processes output.

Claim: The last $n + m - k$ processes output $\leq m$ values.

When the first of those $n + m - k$ processes does final scan S, it sees at most m different pairs. We prove that afterwards,

- Only pairs with those values can appear in 2 locations.
- No other value can ever be output.

Intuition:

$v_5, 5$	$v_3, 3$	$v_2, 2$	$v_2, 2$	$v_2, 2$	$v_3, 3$	$v_5, 5$

Agreement

Agreement: At most k different values are output.

We don't care what the first $k - m$ processes output.

Claim: The last $n + m - k$ processes output $\leq m$ values.

When the first of those $n + m - k$ processes does final scan S, it sees at most m different pairs. We prove that afterwards,

- Only pairs with those values can appear in 2 locations.
- No other value can ever be output.

Intuition:

$\longleftarrow n + 2m - k \longrightarrow$

$v_5, 5$	$v_3, 3$	$v_2, 2$	$v_2, 2$	$v_2, 2$	$v_3, 3$	$v_5, 5$

Agreement

Agreement: At most k different values are output.

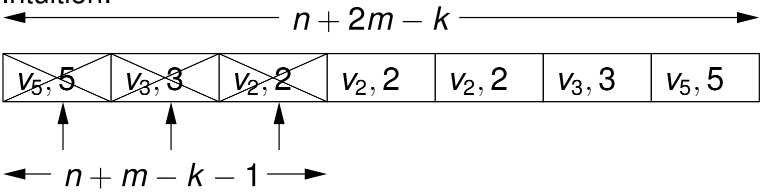
We don't care what the first $k - m$ processes output.

Claim: The last $n + m - k$ processes output $\leq m$ values.

When the first of those $n + m - k$ processes does final scan S, it sees at most m different pairs. We prove that afterwards,

- Only pairs with those values can appear in 2 locations.
- No other value can ever be output.

Intuition:



Agreement

Agreement: At most k different values are output.

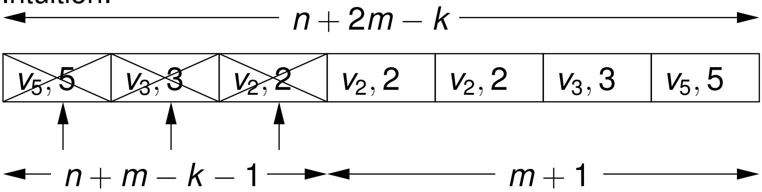
We don't care what the first $k - m$ processes output.

Claim: The last $n + m - k$ processes output $\leq m$ values.

When the first of those $n + m - k$ processes does final scan S, it sees at most m different pairs. We prove that afterwards,

- Only pairs with those values can appear in 2 locations.
- No other value can ever be output.

Intuition:



m-Obstruction-Free Termination

Termination: If at most m processes continue taking steps, one will terminate.

A process stops when it sees at most m different pairs in A.

If at most m processes continue taking steps, we prove

- They cannot all stand still (exchanging preferences)
- Eventually only their pairs are stored in A
- Two pairs with same id have same value

$\Rightarrow$ m-obstruction-free termination

YORK UNIVERSITÉ UNIVERSITY

m-Obstruction-Free Termination

Termination: If at most m processes continue taking steps, one will terminate.

A process stops when it sees at most m different pairs in A.

If at most m processes continue taking steps, we prove

- They cannot all stand still (exchanging preferences)
- Eventually only their pairs are stored in A
- Two pairs with same id have same value

$\Rightarrow$ m-obstruction-free termination

m-Obstruction-Free Termination

Termination: If at most *m* processes continue taking steps, one will terminate.

A process stops when it sees at most *m* different pairs in *A*.

If at most *m* processes continue taking steps, we prove

- They cannot all stand still (exchanging preferences)
- Eventually only their pairs are stored in *A*
- Two pairs with same id have same value

⇒ *m*-obstruction-free termination

m-Obstruction-Free Termination

Termination: If at most m processes continue taking steps, one will terminate.

A process stops when it sees at most m different pairs in A.

If at most m processes continue taking steps, we prove

- They cannot all stand still (exchanging preferences)
- Eventually only their pairs are stored in A
- Two pairs with same id have same value

$\Rightarrow$ m-obstruction-free termination

YORK UNIVERSITÉ UNIVERSITY

m-Obstruction-Free Termination

Termination: If at most m processes continue taking steps, one will terminate.

A process stops when it sees at most m different pairs in A.

If at most m processes continue taking steps, we prove

- They cannot all stand still (exchanging preferences)
- Eventually only their pairs are stored in A
- Two pairs with same id have same value

$\Rightarrow$ m-obstruction-free termination

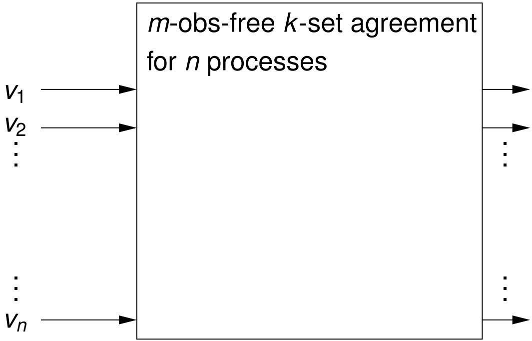
Algorithm For Repeated Set Agreement

Main idea:

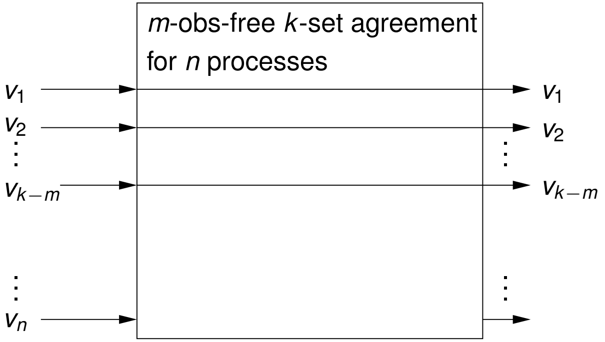
- Write history of output values for all previous instances, along with *id* and *pref*.
- Ignore entries written by processes working on earlier instances.
- If you read value written by a process working on a later instance, adopt its output for your instance.

Can be done using the same number of registers.

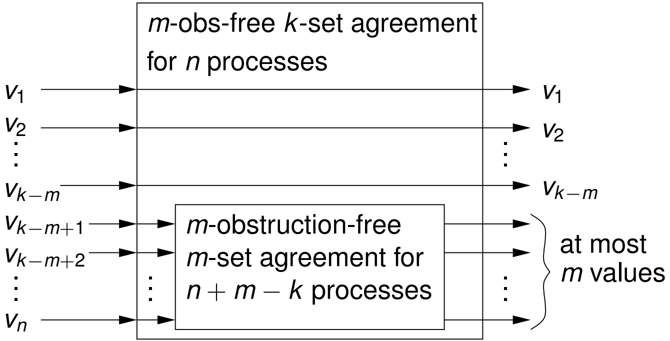
Simpler Algorithm When Ids Are Known



Simpler Algorithm When Ids Are Known



Simpler Algorithm When Ids Are Known



Anonymous Lower Bound

For any set V of m input values
let $\alpha(V)$ be a run of m processes that outputs those m values.

We consider the sequence of registers written (for the first time) in $\alpha(V)$.

Claim

If $r \leq \sqrt{m(\frac{n}{k} - 2)}$ there are infinitely many sets V such that $\alpha(V)$ writes to the same sequence of $r + 1$ registers.

Yields the $\Omega(\sqrt{\frac{mn}{k}})$ lower bound.

Anonymous Lower Bound

For any set V of m input values
let $\alpha(V)$ be a run of m processes that outputs those m values.

We consider the sequence of registers written (for the first time) in $\alpha(V)$.

Claim

If $r \leq \sqrt{m(\frac{n}{k} - 2)}$ there are infinitely many sets V such that $\alpha(V)$ writes to the same sequence of $r + 1$ registers.

Yields the $\Omega(\sqrt{\frac{mn}{k}})$ lower bound.

Proof of Claim

Inductively construct register sequence $\mathbf{R}_i$ of length i such that infinitely many $\alpha(V)$'s register sequences start with $\mathbf{R}_i$.

$\mathbf{R}_0 = \langle\rangle$.

Suppose we have $\mathbf{R}_{i-1}$.

Consider V's such that $\alpha(V)$'s register sequence starts with $\mathbf{R}_{i-1}$.
If there are $\frac{k+1}{m}$ disjoint V's such that $\alpha(V)$ writes only to $\mathbf{R}_{i-1}$, combine them to get run with $k+1$ outputs. Contradiction.

So infinitely many of the V's have longer register sequence.
One register R appears next in infinitely many of the V's register sequences.
Take $\mathbf{R}_i = \mathbf{R}_{i-1} \cdot \langle R \rangle$.

Proof of Claim

Inductively construct register sequence $\mathbf{R}_i$ of length i such that infinitely many $\alpha(V)$'s register sequences start with $\mathbf{R}_i$.

$\mathbf{R}_0 = \langle\rangle$.

Suppose we have $\mathbf{R}_{i-1}$.

Consider V's such that $\alpha(V)$'s register sequence starts with $\mathbf{R}_{i-1}$.
If there are $\frac{k+1}{m}$ disjoint V's such that $\alpha(V)$ writes only to $\mathbf{R}_{i-1}$, combine them to get run with $k+1$ outputs. Contradiction.

So infinitely many of the V's have longer register sequence.
One register R appears next in infinitely many of the V's register sequences.
Take $\mathbf{R}_i = \mathbf{R}_{i-1} \cdot \langle R \rangle$.

Proof of Claim

Inductively construct register sequence $\mathbf{R}_i$ of length i such that infinitely many $\alpha(V)$'s register sequences start with $\mathbf{R}_i$.

$\mathbf{R}_0 = \langle\rangle$.

Suppose we have $\mathbf{R}_{i-1}$.

Consider V's such that $\alpha(V)$'s register sequence starts with $\mathbf{R}_{i-1}$.
If there are $\frac{k+1}{m}$ disjoint V's such that $\alpha(V)$ writes only to $\mathbf{R}_{i-1}$, combine them to get run with $k+1$ outputs. Contradiction.

So infinitely many of the V's have longer register sequence.
One register R appears next in infinitely many of the V's register sequences.
Take $\mathbf{R}_i = \mathbf{R}_{i-1} \cdot \langle R \rangle$.

Proof of Claim

Inductively construct register sequence $\mathbf{R}_i$ of length i such that infinitely many $\alpha(V)$'s register sequences start with $\mathbf{R}_i$.

$\mathbf{R}_0 = \langle\rangle$.

Suppose we have $\mathbf{R}_{i-1}$.

Consider V's such that $\alpha(V)$'s register sequence starts with $\mathbf{R}_{i-1}$. If there are $\frac{k+1}{m}$ disjoint V's such that $\alpha(V)$ writes only to $\mathbf{R}_{i-1}$, combine them to get run with $k+1$ outputs. Contradiction.

So infinitely many of the V's have longer register sequence. One register R appears next in infinitely many of the V's register sequences.
Take $\mathbf{R}_i = \mathbf{R}_{i-1} \cdot \langle R \rangle$.

Proof of Claim

Inductively construct register sequence $\mathbf{R}_i$ of length i such that infinitely many $\alpha(V)$'s register sequences start with $\mathbf{R}_i$.

$\mathbf{R}_0 = \langle\rangle$.

Suppose we have $\mathbf{R}_{i-1}$.

Consider V's such that $\alpha(V)$'s register sequence starts with $\mathbf{R}_{i-1}$.
If there are $\frac{k+1}{m}$ disjoint V's such that $\alpha(V)$ writes only to $\mathbf{R}_{i-1}$, combine them to get run with $k+1$ outputs. Contradiction.
Combined run uses $\Theta(\frac{r^2 k}{m})$ processes,
so can continue this argument as long as r is $O(\sqrt{\frac{nm}{k}})$.
So infinitely many of the V's have longer register sequence.
One register R appears next in infinitely many of the V's register sequences.
Take $\mathbf{R}_i = \mathbf{R}_{i-1} \cdot \langle R \rangle$.

Recap

Bounds on number of registers needed for
m-obstruction-free k-set agreement for n processes

	Repeated	One-Shot
Non-Anon.	$\geq n+m-k$ $\leq n+2m-k$ $\leq n+m-k$ (known ids)	≥ 2 [DFGR13] $\leq n+2m-k$ $\leq n+m-k$ (known ids)
Anon.	$\geq n+m-k$ $\leq (m+1)(n-k)+m^2+1$	$\geq \sqrt{m(\frac{n}{k}-2)}$ $\leq (m+1)(n-k)+m^2$

Recap

Bounds on number of registers needed for
m-obstruction-free k-set agreement for n processes

	Repeated	One-Shot
Non-Anon.	$\geq n+m-k$ $\leq n+2m-k$ $\leq n+m-k$ (known ids)	≥ 2 [DFGR13] $\leq n+2m-k$ $\leq n+m-k$ (known ids)
Anon.	$\geq n+m-k$ $\leq (m+1)(n-k)+m^2+1$	$\geq \sqrt{m(\frac{n}{k}-2)}$ $\leq (m+1)(n-k)+m^2$

Open Problems: Close the gaps.