How synchronizing are primitive groups?

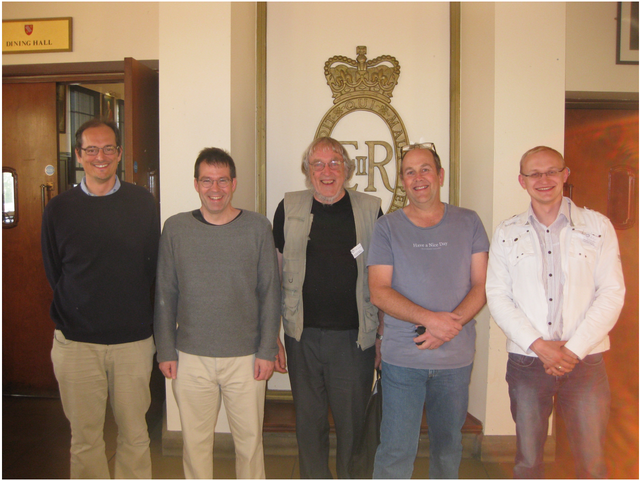
Wolfram Bentz

CEMAT/CIÊNCIAS, Universidade de Lisboa

Joint work with
João Araújo (CEMAT/CIÊNCIAS, Universidade de Lisboa)
Peter J. Cameron (Mathematical Institute, University of St Andrews)
Gordon Royle (Centre for the Mathematics of Symmetry and Computation, University of Western Australia)
Arthur Schaefer (Mathematical Institute, University of St Andrews)

London Mathematical Society – EPSRC Durham Symposium

July 27, 2015


DINING HALL
Have a Nice Day

The prisoner in the dungeon

Imagine the following:

The prisoner in the dungeon

Imagine the following:

- You are a prisoner.

The prisoner in the dungeon

Imagine the following:

- You are a prisoner.
- You are trapped in a dungeon consisting of identical looking caves.

The prisoner in the dungeon

Imagine the following:

- You are a prisoner.
- You are trapped in a dungeon consisting of identical looking caves.
- Each cave has a red, blue, and yellow door.

The prisoner in the dungeon

Imagine the following:

- You are a prisoner.
- You are trapped in a dungeon consisting of identical looking caves.
- Each cave has a red, blue, and yellow door.
- Behind the red and blue doors are long winding red and blue colored one-way corridors leading to other caves.

The prisoner in the dungeon

Imagine the following:

- You are a prisoner.
- You are trapped in a dungeon consisting of identical looking caves.
- Each cave has a red, blue, and yellow door.
- Behind the red and blue doors are long winding red and blue colored one-way corridors leading to other caves.
- The yellow door in one of the caves leads to freedom, while opening any other yellow door leads to instant death.

The prisoner in the dungeon

Imagine the following:

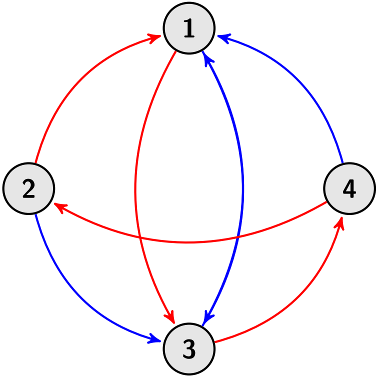
- You are a prisoner.
- You are trapped in a dungeon consisting of identical looking caves.
- Each cave has a red, blue, and yellow door.
- Behind the red and blue doors are long winding red and blue colored one-way corridors leading to other caves.
- The yellow door in one of the caves leads to freedom, while opening any other yellow door leads to instant death.
- You have a complete map of the dungeon, that also shows the favorable yellow door.

The prisoner in the dungeon

Imagine the following:

- You are a prisoner.
- You are trapped in a dungeon consisting of identical looking caves.
- Each cave has a red, blue, and yellow door.
- Behind the red and blue doors are long winding red and blue colored one-way corridors leading to other caves.
- The yellow door in one of the caves leads to freedom, while opening any other yellow door leads to instant death.
- You have a complete map of the dungeon, that also shows the favorable yellow door.
- You do not know which cave you are in.

The prisoner in the dungeon



The prisoner in the dungeon

The prisoner in the dungeon

- What is needed is a method that can guarantee that you end up in the same cave, no matter where you start.

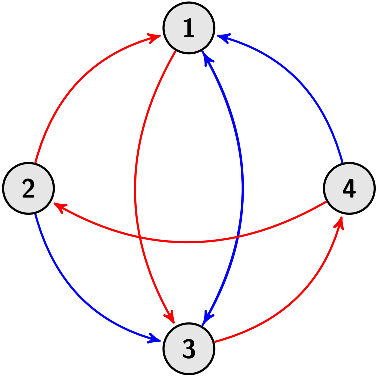
The prisoner in the dungeon

- What is needed is a method that can guarantee that you end up in the same cave, no matter where you start.
- So you want to run a route that will always end up in the same spot.

The prisoner in the dungeon

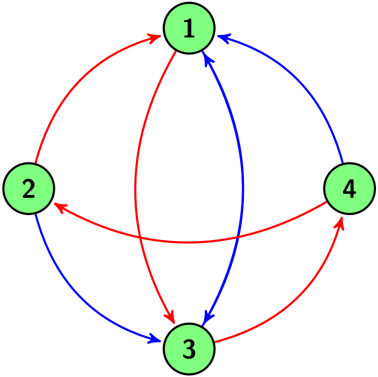
- What is needed is a method that can guarantee that you end up in the same cave, no matter where you start.
- So you want to run a route that will always end up in the same spot.
- Once you know your location you can go wherever you want (if everything is connected).

The prisoner in the dungeon

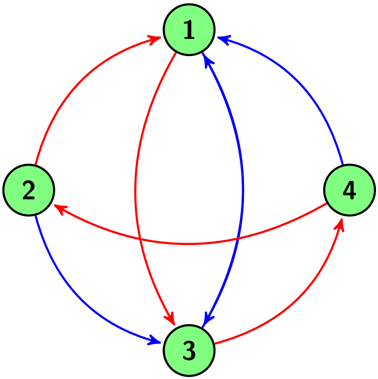


The prisoner in the dungeon

The prisoner in the dungeon

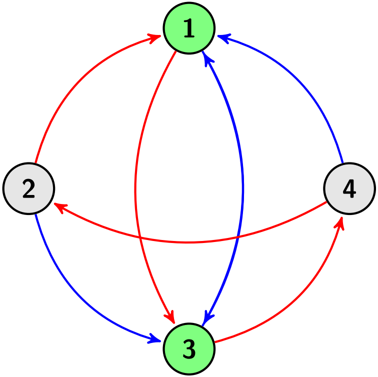


The prisoner in the dungeon



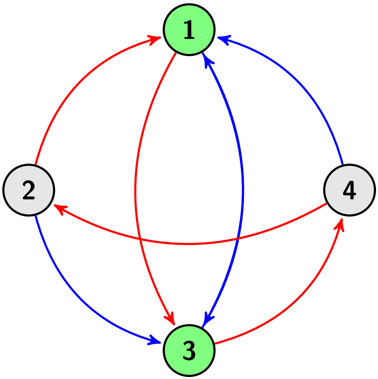
The magic words are BLUE

The prisoner in the dungeon



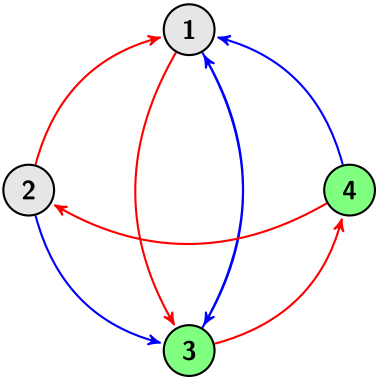
The magic words are BLUE

The prisoner in the dungeon



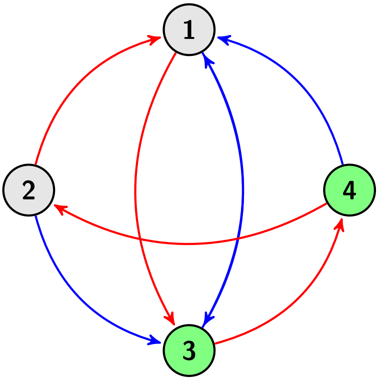
The magic words are BLUE RED

The prisoner in the dungeon



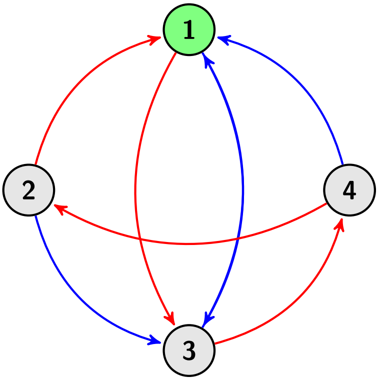
The magic words are BLUE RED

The prisoner in the dungeon



The magic words are BLUE RED BLUE

The prisoner in the dungeon



The magic words are BLUE RED BLUE

Error recovery

Error recovery

- Software (or other systems) tend to enter faulty states from time to time due to unforeseen circumstances.

Error recovery

- Software (or other systems) tend to enter faulty states from time to time due to unforeseen circumstances.
- The standard reaction to such errors can be considered versions of *backward recovery*.

Error recovery

- Software (or other systems) tend to enter faulty states from time to time due to unforeseen circumstances.
- The standard reaction to such errors can be considered versions of *backward recovery*.
- POWER OFF, REBOOT, RESTART FROM THE LAST SAVED CHECKPOINT.

Error recovery

- Software (or other systems) tend to enter faulty states from time to time due to unforeseen circumstances.
- The standard reaction to such errors can be considered versions of *backward recovery*.
- POWER OFF, REBOOT, RESTART FROM THE LAST SAVED CHECKPOINT.
- Such an option might not always exist or be ideal.

- Benenson, Paz-Elizur, Adar, Keinen, Livneh and Shapiro: Programmable and autonomous computing machine made of biomolecules. *Nature, 2011*

- Benenson, Paz-Elizur, Adar, Keinen, Livneh and Shapiro: Programmable and autonomous computing machine made of biomolecules. *Nature, 2011*
- No OFF-button!

- Benenson, Paz-Elizur, Adar, Keinen, Livneh and Shapiro: Programmable and autonomous computing machine made of biomolecules. *Nature, 2011*
- No OFF-button!
- Consider a satellite that has reemerged (say from behind Mars) and is in an unknown orientation.

- Benenson, Paz-Elizur, Adar, Keinen, Livneh and Shapiro: Programmable and autonomous computing machine made of biomolecules. *Nature, 2011*
- No OFF-button!
- Consider a satellite that has reemerged (say from behind Mars) and is in an unknown orientation.
- Not practical to go there, unplug, reorient and start-up again.

- Benenson, Paz-Elizur, Adar, Keinen, Livneh and Shapiro: Programmable and autonomous computing machine made of biomolecules. *Nature, 2011*
- No OFF-button!
- Consider a satellite that has reemerged (say from behind Mars) and is in an unknown orientation.
- Not practical to go there, unplug, reorient and start-up again.
- Consider production tools or other products dropped on a conveyor belt in unknown orientation.

- Benenson, Paz-Elizur, Adar, Keinen, Livneh and Shapiro: Programmable and autonomous computing machine made of biomolecules. *Nature, 2011*
- No OFF-button!
- Consider a satellite that has reemerged (say from behind Mars) and is in an unknown orientation.
- Not practical to go there, unplug, reorient and start-up again.
- Consider production tools or other products dropped on a conveyor belt in unknown orientation.
- Before automatic work can be performed on them, the need to be aligned correctly.

Forward Error Recovery

- An alternative to dealing with errors utilizes *forward recovery*.

Forward Error Recovery

- An alternative to dealing with errors utilizes *forward recovery*.
- A forward recovery mechanism is a sequence of procedures or instructions that brings the process to a known state *irrespectively of its current state*.

Synchronization for Automata, Semigroups, and Groups

Definition

An automaton is *synchronizing* if there is a sequence of inputs that puts the automaton into a specific state, irrespective of its current state.

Definition

S_n and T_n are, respectively, the symmetric group and the full transformation monoid on the set $X = \{1, \ldots, n\}$.
A set $S \subseteq T_X$ of transpositions is *synchronizing* if the semigroup $\langle S \rangle$ contain a constant map.
We say that a group $G \leq S_n$ *synchronizes* a transformation $t \in T_n \setminus S_n$ if the subsemigroup of T_n generated by $G \cup \{t\}$ contains a constant map.

Synchronizing Groups

Definition

A subgroup G of S_n is a *synchronizing group* if it synchronizes every non-permutation in T_n.

Synchronizing Groups

Definition

A subgroup G of S_n is a *synchronizing group* if it synchronizes every non-permutation in T_n.

"Pure" group theoretic definition: A permutation group G on X is synchronizing if no (proper, non-trivial) partition of X has a section (or transversal) S that is invariant under G (i.e. for which S^g is also a section for all $g \in G$)

Synchronizing Groups

Definition

A subgroup G of S_n is a *synchronizing group* if it synchronizes every non-permutation in T_n.

"Pure" group theoretic definition: A permutation group G on X is synchronizing if no (proper, non-trivial) partition of X has a section (or transversal) S that is invariant under G (i.e. for which S^g is also a section for all $g \in G$)

Results on synchronizing groups

Results on synchronizing groups

Definition

A subgroup G of S_X is *primitive* if it acts transitively on X and does not preserve any non-trivial partition of X.

Results on synchronizing groups

Definition

A subgroup G of S_X is *primitive* if it acts transitively on X and does not preserve any non-trivial partition of X.

- Synchronizing groups must be *primitive* ([Araújo 2006], [Arnold and Steinberg 2006], [Neumann 2009]).

Results on synchronizing groups

Definition

A subgroup G of S_X is *primitive* if it acts transitively on X and does not preserve any non-trivial partition of X.

- Synchronizing groups must be *primitive* ([Araújo 2006], [Arnold and Steinberg 2006], [Neumann 2009]).
- If the group G is primitive, but not synchronizing, let t be a transformation not synchronized by G with minimal rank. Then t has *uniform kernel*, i.e. all kernel classes are of the same size ([Neumann 2009]).

Results on synchronizing groups

Definition

A subgroup G of S_X is *primitive* if it acts transitively on X and does not preserve any non-trivial partition of X.

- Synchronizing groups must be *primitive* ([Araújo 2006], [Arnold and Steinberg 2006], [Neumann 2009]).
- If the group G is primitive, but not synchronizing, let t be a transformation not synchronized by G with minimal rank. Then t has *uniform kernel*, i.e. all kernel classes are of the same size ([Neumann 2009]).
- If $|X|$ is prime, every primitive group over X is synchronizing.

From transformation semigroups to graphs

Let $M \leq T_X$, then the graph $\mathrm{Gr}(M)$ has vertex set X and two vertices $x, y \in X$ are adjacent if there is NO $f \in M$ for which $xf = yf$.

From transformation semigroups to graphs

Let $M \leq T_X$, then the graph $\mathrm{Gr}(M)$ has vertex set X and two vertices $x, y \in X$ are adjacent if there is NO $f \in M$ for which $xf = yf$.

Consequences

- $S \subseteq T_X$ defines a synchronizing automaton if and only if $\mathrm{Gr}(\langle S \rangle)$ is the null graph.

From transformation semigroups to graphs

Let $M \leq T_X$, then the graph $\mathrm{Gr}(M)$ has vertex set X and two vertices $x, y \in X$ are adjacent if there is NO $f \in M$ for which $xf = yf$.

Consequences

- $S \subseteq T_X$ defines a synchronizing automaton if and only if $\mathrm{Gr}(\langle S \rangle)$ is the null graph.
- For any $M \leq T_X$, $M \leq \mathrm{End}(\mathrm{Gr}(M))$.

From transformation semigroups to graphs

Let $M \leq T_X$, then the graph $\mathrm{Gr}(M)$ has vertex set X and two vertices $x, y \in X$ are adjacent if there is NO $f \in M$ for which $xf = yf$.

Consequences

- $S \subseteq T_X$ defines a synchronizing automaton if and only if $\mathrm{Gr}(\langle S \rangle)$ is the null graph.
- For any $M \leq T_X$, $M \leq \mathrm{End}(\mathrm{Gr}(M))$.
- For any $M \leq T_X$, the following are equal:

From transformation semigroups to graphs

Let $M \leq T_X$, then the graph Gr(M) has vertex set X and two vertices $x, y \in X$ are adjacent if there is NO $f \in M$ for which $xf = yf$.

Consequences

- $S \subseteq T_X$ defines a synchronizing automaton if and only if $\mathrm{Gr}(\langle S \rangle)$ is the null graph.
- For any $M \leq T_X$, $M \leq \mathrm{End}(\mathrm{Gr}(M))$.
- For any $M \leq T_X$, the following are equal:
    - the smallest rank of an element of M,

From transformation semigroups to graphs

Let $M \leq T_X$, then the graph $\mathrm{Gr}(M)$ has vertex set X and two vertices $x, y \in X$ are adjacent if there is NO $f \in M$ for which $xf = yf$.

Consequences

- $S \subseteq T_X$ defines a synchronizing automaton if and only if $\mathrm{Gr}(\langle S \rangle)$ is the null graph.
- For any $M \leq T_X$, $M \leq \mathrm{End}(\mathrm{Gr}(M))$.
- For any $M \leq T_X$, the following are equal:
  - the smallest rank of an element of M,
  - the clique number of $\mathrm{Gr}(M)$,

From transformation semigroups to graphs

Let $M \leq T_X$, then the graph $\mathrm{Gr}(M)$ has vertex set X and two vertices $x, y \in X$ are adjacent if there is NO $f \in M$ for which $xf = yf$.

Consequences

- $S \subseteq T_X$ defines a synchronizing automaton if and only if $\mathrm{Gr}(\langle S \rangle)$ is the null graph.
- For any $M \leq T_X$, $M \leq \mathrm{End}(\mathrm{Gr}(M))$.
- For any $M \leq T_X$, the following are equal:
  - the smallest rank of an element of M,
  - the clique number of $\mathrm{Gr}(M)$,
  - the chromatic number of $\mathrm{Gr}(M)$.

From transformation semigroups to graphs

Theorem

A transformation semigroup M is non-synchronizing if and only if there is a non-null graph Γ such that $M \subseteq \mathrm{End}(\Gamma)$. Moreover, Γ can be assumed to have coinciding clique and chromatic numbers.

Primitive groups

We will now assume that we generate M from a primitive group and a singular map. This restricts $\mathrm{Gr}(M)$.

Primitive groups

We will now assume that we generate M from a primitive group and a singular map. This restricts $\mathrm{Gr}(M)$.

Let G be a graph with primitive automorphism group on n vertices and clique number r equal to its chromatic number. Then

- G is regular, i.e. all vertices have the same number of neighbors.

Primitive groups

We will now assume that we generate M from a primitive group and a singular map. This restricts $\mathrm{Gr}(M)$.

Let G be a graph with primitive automorphism group on n vertices and clique number r equal to its chromatic number. Then

- G is regular, i.e. all vertices have the same number of neighbors.
- G does not contain an induced subgraph that is equal to a complete graph on $r+1$ vertices with one edge removed (Araújo, Cameron).

Primitive groups

We will now assume that we generate M from a primitive group and a singular map. This restricts $\mathrm{Gr}(M)$.

Let G be a graph with primitive automorphism group on n vertices and clique number r equal to its chromatic number. Then

- G is regular, i.e. all vertices have the same number of neighbors.
- G does not contain an induced subgraph that is equal to a complete graph on $r+1$ vertices with one edge removed (Araújo, Cameron).
- Let l be the number of neighbors of one (and hence every) vertex. For any distinct vertices x, y, their neighborhoods intersect in at most $l-2$ points (also independently by Spiga/Verret)

Primitive groups

We will now assume that we generate M from a primitive group and a singular map. This restricts $\mathrm{Gr}(M)$.

Let G be a graph with primitive automorphism group on n vertices and clique number r equal to its chromatic number. Then

- G is regular, i.e. all vertices have the same number of neighbors.
- G does not contain an induced subgraph that is equal to a complete graph on $r+1$ vertices with one edge removed (Araújo, Cameron).
- Let l be the number of neighbors of one (and hence every) vertex. For any distinct vertices x, y, their neighborhoods intersect in at most $l-2$ points (also independently by Spiga/Verret)
- Suppose that A is a set of vertices, such that the union of their neighborhoods has size $l+d$. Then $|A| \leq \binom{d+1}{2}$.

Primitive groups and maps of high rank

It was previously known that a primitive group $G \leq S_n$ synchronizes every map of rank $n-1$ (Rystsov) and $n-2$ (Araújo, Cameron).

Theorem

A primitive group $G \leq S_n$ synchronizes every map of rank $n-3$ and $n-4$.

Primitive groups and maps of high rank

It was previously known that a primitive group $G \leq S_n$ synchronizes every map of rank $n-1$ (Rystsov) and $n-2$ (Araújo, Cameron).

Theorem

A primitive group $G \leq S_n$ synchronizes every map of rank $n-3$ and $n-4$.

The theorem does not generalize to $n-6$. Hence either maps of rank $n-5$ are always synchronized or $n-4$ is best possible.

Primitive groups and maps of high rank

It was previously known that a primitive group $G \leq S_n$ synchronizes every map of rank $n-1$ (Rystsov) and $n-2$ (Araújo, Cameron).

Theorem

A primitive group $G \leq S_n$ synchronizes every map of rank $n-3$ and $n-4$.

The theorem does not generalize to $n-6$. Hence either maps of rank $n-5$ are always synchronized or $n-4$ is best possible.

Some examples from the proof:

- The base case: kernel type $(4, 1, \ldots, 1)$ (Araújo, Cameron)

Primitive groups and maps of high rank

It was previously known that a primitive group $G \leq S_n$ synchronizes every map of rank $n-1$ (Rystsov) and $n-2$ (Araújo, Cameron).

Theorem

A primitive group $G \leq S_n$ synchronizes every map of rank $n-3$ and $n-4$.

The theorem does not generalize to $n-6$. Hence either maps of rank $n-5$ are always synchronized or $n-4$ is best possible.

Some examples from the proof:

- The base case: kernel type $(4, 1, \ldots, 1)$ (Araújo, Cameron)
- The good case: kernel type $(3, 2, 1, \ldots, 1)$ (also independently by Spiga/Verret)

Primitive groups and maps of high rank

It was previously known that a primitive group $G \leq S_n$ synchronizes every map of rank $n-1$ (Rystsov) and $n-2$ (Araújo, Cameron).

Theorem

A primitive group $G \leq S_n$ synchronizes every map of rank $n-3$ and $n-4$.

The theorem does not generalize to $n-6$. Hence either maps of rank $n-5$ are always synchronized or $n-4$ is best possible.

Some examples from the proof:

- The base case: kernel type $(4, 1, \dots, 1)$ (Araújo, Cameron)
- The good case: kernel type $(3, 2, 1, \dots, 1)$ (also independently by Spiga/Verret)
- Reduction to a computational problem: kernel type $(2, 2, 2, 1, \dots, 1)$

Primitive groups and maps of high rank

It was previously known that a primitive group $G \leq S_n$ synchronizes every map of rank $n-1$ (Rystsov) and $n-2$ (Araújo, Cameron).

Theorem

A primitive group $G \leq S_n$ synchronizes every map of rank $n-3$ and $n-4$.

The theorem does not generalize to $n-6$. Hence either maps of rank $n-5$ are always synchronized or $n-4$ is best possible.

Some examples from the proof:

- The base case: kernel type $(4, 1, \ldots, 1)$ (Araújo, Cameron)
- The good case: kernel type $(3, 2, 1, \ldots, 1)$ (also independently by Spiga/Verret)
- Reduction to a computational problem: kernel type $(2, 2, 2, 1, \ldots, 1)$
- Rank $n-4$: many cases, tricky cases!

Rank n-4

Rank n-4

Case: Kernel type $(3, 2, 2, 1, \ldots, 1)$

Rank n-4

Case: Kernel type $(3, 2, 2, 1, \ldots, 1)$
Subcase: the images of the non-trivial kernel classes form a triangle

Rank n-4

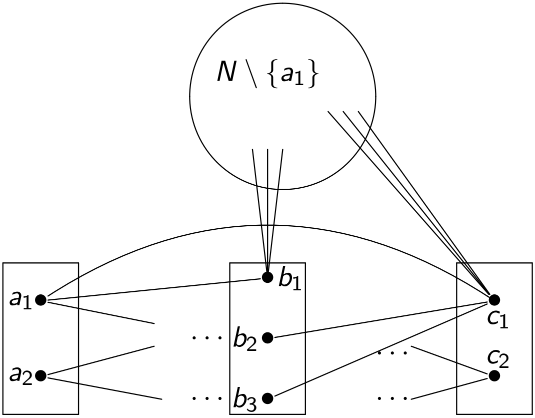
Case: Kernel type $(3, 2, 2, 1, \ldots, 1)$
Subcase: the images of the non-trivial kernel classes form a triangle
Subsubcase 1:

Rank n-4

Case: Kernel type $(3, 2, 2, 1, \dots, 1)$

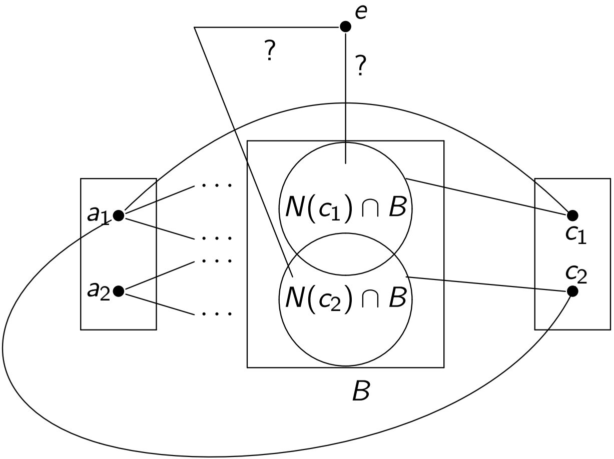
Subcase: the images of the non-trivial kernel classes form a triangle

Subsubcase 1:



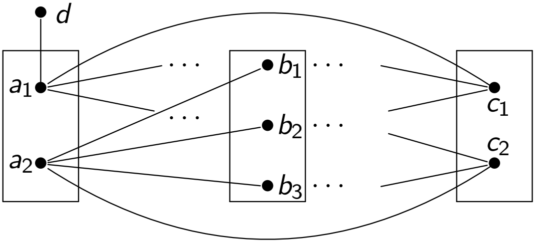
Rank n-4

Case: Kernel type $(3, 2, 2, 1, \ldots, 1)$
Subcase: the images of the non-trivial kernel classes form a triangle
Subsubcase 2:



Rank n-4

Case: Kernel type $(3, 2, 2, 1, \ldots, 1)$
Subcase: the images of the non-trivial kernel classes form a triangle
Subsubcase 3:



More results for maps of large rank

- Suppose that $j \geq 1$, $d_1, \ldots, d_j \geq 2$, $d = -j + \Sigma d_i$, $l \geq \binom{d+1}{2} + 1$, and G a primitive group. Then G synchronized every map of kernel type $(l, d_1, \ldots, d_j, 1, \ldots, 1)$.

More results for maps of large rank

- Suppose that $j \geq 1$, $d_1, \ldots, d_j \geq 2$, $d = -j + \Sigma d_i$, $l \geq \binom{d+1}{2} + 1$, and G a primitive group. Then G synchronized every map of kernel type $(l, d_1, \ldots, d_j, 1, \ldots, 1)$.
- Let G act primitively on X with $|X| = n$ and suppose that the induced action of G on X^2 has exactly three orbits. Then G synchronizes every map of rank larger than $n - \left(1 + \sqrt{n-1}/12\right)$.

Maps of small ranks - Flashback

Definition (Araújo, WB, Cameron (ABC))

A subgroup G of S_n is an *almost synchronizing group* if it synchronizes every transformation of non-uniform kernel in T_n.

Maps of small ranks - Flashback

Definition (Araújo, WB, Cameron (ABC))

A subgroup G of S_n is an *almost synchronizing group* if it synchronizes every transformation of non-uniform kernel in T_n.

Maps of small ranks - Flashback

Definition (Araújo, WB, Cameron (ABC))

A subgroup G of S_n is an *almost synchronizing group* if it synchronizes every transformation of non-uniform kernel in T_n.

We have developed two different methods to recognize almost synchronizing groups. Both give an infinite list of examples (with little overlap).

Maps of small ranks - Flashback

Definition (Araújo, WB, Cameron (ABC))

A subgroup G of S_n is an *almost synchronizing group* if it synchronizes every transformation of non-uniform kernel in T_n.

We have developed two different methods to recognize almost synchronizing groups. Both give an infinite list of examples (with little overlap).

Conjecture

Every primitive group is almost synchronizing.

Maps of small ranks - Flashback

Definition (Araújo, WB, Cameron (ABC))

A subgroup G of S_n is an *almost synchronizing group* if it synchronizes every transformation of non-uniform kernel in T_n.

We have developed two different methods to recognize almost synchronizing groups. Both give an infinite list of examples (with little overlap).

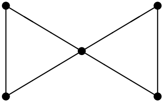
Conjecture

Every primitive group is almost synchronizing.

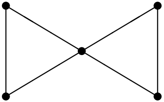
Theorem (Araújo, Cameron)

Every primitive groups synchronizes every non-uniform map of rank at most 4*.*

The tale of the butterfly



The tale of the butterfly



Theorem

There exists a primitive group G acting on a set X of 45 points and a transformation t on X, such that G does not synchronize t and t has kernel type $(15, 10, 10, 5, 5)$.

How synchronizing are primitive groups?

- The "typical" primitive group is synchronizing.

How synchronizing are primitive groups?

- The "typical" primitive group is synchronizing.
- Primitive groups are synchronizing for maps of large rank.

How synchronizing are primitive groups?

- The "typical" primitive group is synchronizing.
- Primitive groups are synchronizing for maps of large rank.
- There are cases of primitive groups with "synchronization deficiencies".

How synchronizing are primitive groups?

- The "typical" primitive group is synchronizing.
- Primitive groups are synchronizing for maps of large rank.
- There are cases of primitive groups with "synchronization deficiencies".
- How deficient can this get in the worst case?

How synchronizing are primitive groups?

- The "typical" primitive group is synchronizing.
- Primitive groups are synchronizing for maps of large rank.
- There are cases of primitive groups with "synchronization deficiencies".
- How deficient can this get in the worst case?

How synchronizing are primitive groups?

- The "typical" primitive group is synchronizing.
- Primitive groups are synchronizing for maps of large rank.
- There are cases of primitive groups with "synchronization deficiencies".
- How deficient can this get in the worst case? See Gordon's talk!

A final conjecture

Conjecture

A primitive group of rank n synchronizes every map of rank r for $n/2 < r < n$.

A final conjecture

Conjecture

A primitive group of rank n synchronizes every map of rank r for $n/2 < r < n$.

Thank you!

J. Araújo, A group theoretical approach to synchronizing automata and the Černy problem.Unpublished manuscript, 2006.

J. Araújo, W. Bentz and P.J. Cameron, Groups Synchronizing a Transformation of Non-Uniform Kernel. *Theoret. Comput. Sci.*, **498** (2013), 1–9.

J. Araújo and P. J. Cameron, Primitive Groups Synchronize Non-uniform Maps of Extreme Ranks, *Journal of Combinatorial Theorie, Series B*, **106** (2014), 98–114.

F. Arnold and B. Steinberg, Synchronizing groups and automata. *Theoret. Comput. Sci.* **359** (2006), no. 1-3, 101–110.

Y. Benenson, T. Paz-Elizur, R. Adar, E. Keinan, Z. Livneh, and E. Shapiro(2001). Programmable and autonomous computing machine made of biomolecules. *Nature 404(6862)*, 430–434.

I. Rystsov, Quasioptimal bound for the length of reset words for regular automata. *Acta Cybernet.* **12** (1995), no. 2, 145–152.

P. M. Neumann, Primitive permutation groups and their section-regular partitions. *Michigan Math. J.* **58** (2009), 309–322.