Resistance distance and resistance distance transform

Peter J. Cameron, University of St Andrews

CCGTA
Boca Raton, 15 August 2022

Two questions

I want to begin with two questions.

Two questions

I want to begin with two questions.

- Which graphs make the best networks?

Two questions

I want to begin with two questions.

- Which graphs make the best networks?
- How can we tell whether two graphs are the same?

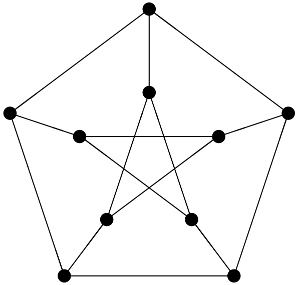
Two questions

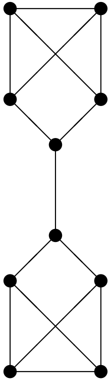
I want to begin with two questions.

- Which graphs make the best networks?
- How can we tell whether two graphs are the same?

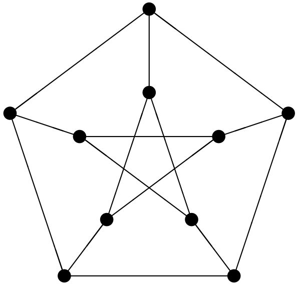
My main interest here is in the second question, but you will see that the two questions are more closely related than you might first think.

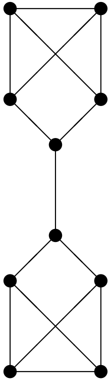
What makes a good network?





What makes a good network?





Suppose you wanted to build a network connecting ten nodes. You could afford to construct fifteen edges. You want the network to be well connected and resilient. Which one of the two shown above would you choose?

I think everyone would agree to choose the first graph. No two nodes are more than two steps apart, whereas the maximum distance in the second is 5.

I think everyone would agree to choose the first graph. No two nodes are more than two steps apart, whereas the maximum distance in the second is 5.
Moreover, the second network has an obvious bottleneck, the edge in the middle; if this fails, the network will fall apart into two parts of size 5. We would need five edges to fail in order to disconnect the first network into two equal parts.

I think everyone would agree to choose the first graph. No two nodes are more than two steps apart, whereas the maximum distance in the second is 5.
Moreover, the second network has an obvious bottleneck, the edge in the middle; if this fails, the network will fall apart into two parts of size 5. We would need five edges to fail in order to disconnect the first network into two equal parts.
The first graph is the celebrated Petersen graph, and has many nice properties including symmetry (its automorphism group has order 120).

I think everyone would agree to choose the first graph. No two nodes are more than two steps apart, whereas the maximum distance in the second is 5.

Moreover, the second network has an obvious bottleneck, the edge in the middle; if this fails, the network will fall apart into two parts of size 5. We would need five edges to fail in order to disconnect the first network into two equal parts.

The first graph is the celebrated Petersen graph, and has many nice properties including symmetry (its automorphism group has order 120).

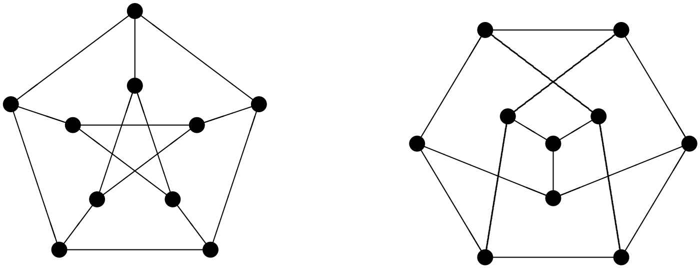
But what if we are faced with huge networks which cannot be drawn in a simple way?

When are two graphs the same?

Two graphs Γ and Δ are isomorphic if there is a bijection from the vertex set of Γ to the vertex set of Δ which carries edges to edges and non-edges to non-edges.

When are two graphs the same?

Two graphs Γ and Δ are isomorphic if there is a bijection from the vertex set of Γ to the vertex set of Δ which carries edges to edges and non-edges to non-edges.
Are these two graphs isomorphic?



Again it is not too difficult to see that they are: just start matching up vertices.

Again it is not too difficult to see that they are: just start matching up vertices.

However, with two large graphs, drawn in an unhelpful way, it may not be easy to decide.

Again it is not too difficult to see that they are: just start matching up vertices.
However, with two large graphs, drawn in an unhelpful way, it may not be easy to decide.
This is the famous graph isomorphism problem, one of a select class of problems in the complexity class NP which are not known to be either in P (polynomial-time solvable) or NP-complete (equivalent to the hardest problems in NP).
In the last decade, László Babai found an algorithm for graph isomorphism which runs in quasipolynomial time, that is, time bounded by $O(\exp(a(\log n)^c)$ for some constants a and c. (This is polynomial if $c = 1$.)

Coherent configurations

One of the pioneers on the graph isomorphism problem was Boris Weisfeiler, who worked on it in the former Soviet Union in the 1960s. He emigrated to the USA in the 1970s, then in the 1980s went hiking in Chile; he disappeared, and no trace of him has ever been found.

Coherent configurations

One of the pioneers on the graph isomorphism problem was Boris Weisfeiler, who worked on it in the former Soviet Union in the 1960s. He emigrated to the USA in the 1970s, then in the 1980s went hiking in Chile; he disappeared, and no trace of him has ever been found.

Weisfeiler and his colleague Leman devised an algorithm which, given a graph, constructs a canonical refinement of it, an object which they called a cellular algebra. If two graphs are isomorphic, then these cellular algebras are isomorphic; since they usually have much more structure, it is simpler in practice to test this.

Coherent configurations

One of the pioneers on the graph isomorphism problem was Boris Weisfeiler, who worked on it in the former Soviet Union in the 1960s. He emigrated to the USA in the 1970s, then in the 1980s went hiking in Chile; he disappeared, and no trace of him has ever been found.

Weisfeiler and his colleague Leman devised an algorithm which, given a graph, constructs a canonical refinement of it, an object which they called a cellular algebra. If two graphs are isomorphic, then these cellular algebras are isomorphic; since they usually have much more structure, it is simpler in practice to test this.

Weisfeiler's ideas are deeply embedded in Babai's proof.

Coherent configurations

The cellular algebras of the Weisfeiler–Leman algorithm are now called coherent configurations.

Coherent configurations

The cellular algebras of the Weisfeiler–Leman algorithm are now called coherent configurations.
In the 1960s, as well as Weisfeiler and colleagues, two other schools came up with similar ideas:

Coherent configurations

The cellular algebras of the Weisfeiler–Leman algorithm are now called coherent configurations.
In the 1960s, as well as Weisfeiler and colleagues, two other schools came up with similar ideas:

- In group theory, Wielandt extended the method of Schur rings invented by Schur. This was refined by Donald Higman under the name coherent configurations.

Coherent configurations

The cellular algebras of the Weisfeiler–Leman algorithm are now called coherent configurations.
In the 1960s, as well as Weisfeiler and colleagues, two other schools came up with similar ideas:

- In group theory, Wielandt extended the method of Schur rings invented by Schur. This was refined by Donald Higman under the name coherent configurations.
- A special case was introduced by Bose and his students in statistics, under the name association schemes. Statisticians need to invert large symmetric matrices; before the days of computers, this was much easier if the matrices were patterned in a certain way.

Coherent configurations

The cellular algebras of the Weisfeiler–Leman algorithm are now called coherent configurations.

In the 1960s, as well as Weisfeiler and colleagues, two other schools came up with similar ideas:

- In group theory, Wielandt extended the method of Schur rings invented by Schur. This was refined by Donald Higman under the name coherent configurations.
- A special case was introduced by Bose and his students in statistics, under the name association schemes. Statisticians need to invert large symmetric matrices; before the days of computers, this was much easier if the matrices were patterned in a certain way.

Now the term "cellular algebra" has been used in a different context, so these objects are now called "coherent configurations".

The definition

A coherent configuration is a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying certain properties, which follow. Associating a colour with each relation, we can think of this as an edge-coloouring of the complete directed graph with loops on Ω.

The definition

A coherent configuration is a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying certain properties, which follow. Associating a colour with each relation, we can think of this as an edge-coloouring of the complete directed graph with loops on Ω.

- Every ordered pair satisfies exactly one of the relations.

The definition

A coherent configuration is a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying certain properties, which follow. Associating a colour with each relation, we can think of this as an edge-coloouring of the complete directed graph with loops on Ω.

- Every ordered pair satisfies exactly one of the relations.
- The relation of equality (the diagonal of $\Omega \times \Omega$) is the union of some of these relations.

The definition

A coherent configuration is a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying certain properties, which follow. Associating a colour with each relation, we can think of this as an edge-coloouring of the complete directed graph with loops on Ω.

- Every ordered pair satisfies exactly one of the relations.
- The relation of equality (the diagonal of $\Omega \times \Omega$) is the union of some of these relations.
- The transpose $\{(y, x) : (x, y) \in R_i\}$ of any relation is another relation in the set.

The definition

A coherent configuration is a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying certain properties, which follow. Associating a colour with each relation, we can think of this as an edge-coloouring of the complete directed graph with loops on Ω.

- Every ordered pair satisfies exactly one of the relations.
- The relation of equality (the diagonal of $\Omega \times \Omega$) is the union of some of these relations.
- The transpose $\{(y, x) : (x, y) \in R_i\}$ of any relation is another relation in the set.
- Given $i, j, k \in \{1, \ldots, r\}$ and $(x, y) \in R_i$, the number of $z \in \Omega$ such that $(x, z) \in R_j$ and $(z, y) \in R_k$ depends only on (i, j, k) and not on the choice of x and y.

The definition

A coherent configuration is a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying certain properties, which follow. Associating a colour with each relation, we can think of this as an edge-coloouring of the complete directed graph with loops on Ω.

- Every ordered pair satisfies exactly one of the relations.
- The relation of equality (the diagonal of $\Omega \times \Omega$) is the union of some of these relations.
- The transpose $\{(y, x) : (x, y) \in R_i\}$ of any relation is another relation in the set.
- Given $i, j, k \in \{1, \ldots, r\}$ and $(x, y) \in R_i$, the number of $z \in \Omega$ such that $(x, z) \in R_j$ and $(z, y) \in R_k$ depends only on (i, j, k) and not on the choice of x and y.

The number r is called the rank of the configuration. If all the relations R_i are symmetric, the configuration is an association scheme.

The Weisfeiler–Leman algorithm

The set of coherent configurations on Ω is closed under join of partitions, and contains the partition into singletons. So, given any partition Π of Ω^2, there is a unique coarsest coherent configuration which refines Π. The WL algorithm finds this configuration. It works as follows.

The Weisfeiler–Leman algorithm

The set of coherent configurations on Ω is closed under join of partitions, and contains the partition into singletons. So, given any partition Π of Ω^2, there is a unique coarsest coherent configuration which refines Π. The WL algorithm finds this configuration. It works as follows.

Given $\Pi = \{R_1, \ldots, R_s\}$, regard it as a collection of edge-coloured digraphs on Ω (one for each relation). Now, for each triple (i, j, k), and each choice of $(x, y) \in R_i$, count the number of z for which $(x, z) \in R_j$ and $(z, y) \in R_k$. In general these numbers will not be constant. So refine the partition by splitting R_i into a number of parts, so that these numbers are constant on each part.

The Weisfeiler–Leman algorithm

The set of coherent configurations on Ω is closed under join of partitions, and contains the partition into singletons. So, given any partition Π of Ω^2, there is a unique coarsest coherent configuration which refines Π. The WL algorithm finds this configuration. It works as follows.

Given $\Pi = \{R_1, \ldots, R_s\}$, regard it as a collection of edge-coloured digraphs on Ω (one for each relation). Now, for each triple (i, j, k), and each choice of $(x, y) \in R_i$, count the number of z for which $(x, z) \in R_j$ and $(z, y) \in R_k$. In general these numbers will not be constant. So refine the partition by splitting R_i into a number of parts, so that these numbers are constant on each part.

Then iterate this construction. It stabilises after finitely many steps, and the stable partition is clearly a coherent configuration.

Another approach

The WL algorithm is simple and efficient. However, it may take a number of steps to reach the stable partition.

Another approach

The WL algorithm is simple and efficient. However, it may take a number of steps to reach the stable partition.
In view of this, Mikhail Klin and Michael Kagan proposed a different refinement algorithm. Their algorithm, to which I now turn, involves regarding the graph or edge partition as an electrical network, and computing the effective resistance between pairs of vertices.

Another approach

The WL algorithm is simple and efficient. However, it may take a number of steps to reach the stable partition.

In view of this, Mikhail Klin and Michael Kagan proposed a different refinement algorithm. Their algorithm, to which I now turn, involves regarding the graph or edge partition as an electrical network, and computing the effective resistance between pairs of vertices.

This idea is also related to the question of good networks, as we will see.

Resistance

Given a graph on the vertex set Ω, we can regard it as an electrical network. The simplest way to do this is to put a 1-ohm resistor on each edge of the graph. Then we can measure the effective resistance between any two vertices by connecting those vertices to a 1-volt battery and measuring the current I which flows: the effective resistance is then $1/I$.

Resistance

Given a graph on the vertex set Ω, we can regard it as an electrical network. The simplest way to do this is to put a 1-ohm resistor on each edge of the graph. Then we can measure the effective resistance between any two vertices by connecting those vertices to a 1-volt battery and measuring the current I which flows: the effective resistance is then $1/I$.

The effective resistance can be calculated by using the relevant physics:

Resistance

Given a graph on the vertex set Ω, we can regard it as an electrical network. The simplest way to do this is to put a 1-ohm resistor on each edge of the graph. Then we can measure the effective resistance between any two vertices by connecting those vertices to a 1-volt battery and measuring the current I which flows: the effective resistance is then $1/I$.
The effective resistance can be calculated by using the relevant physics:

- Kirchhoff's voltage law: the sum of the potential differences along any two paths joining vertices v and w is independent of the chosen paths.

Resistance

Given a graph on the vertex set Ω, we can regard it as an electrical network. The simplest way to do this is to put a 1-ohm resistor on each edge of the graph. Then we can measure the effective resistance between any two vertices by connecting those vertices to a 1-volt battery and measuring the current I which flows: the effective resistance is then $1/I$.

The effective resistance can be calculated by using the relevant physics:

- Kirchhoff's voltage law: the sum of the potential differences along any two paths joining vertices v and w is independent of the chosen paths.
- Kirchhoff's current law: for any vertex v apart from those connected to the battery, the current flowing into v is equal to the current flowing out of v.

Resistance

Given a graph on the vertex set Ω, we can regard it as an electrical network. The simplest way to do this is to put a 1-ohm resistor on each edge of the graph. Then we can measure the effective resistance between any two vertices by connecting those vertices to a 1-volt battery and measuring the current I which flows: the effective resistance is then $1/I$.
The effective resistance can be calculated by using the relevant physics:

- Kirchhoff's voltage law: the sum of the potential differences along any two paths joining vertices v and w is independent of the chosen paths.
- Kirchhoff's current law: for any vertex v apart from those connected to the battery, the current flowing into v is equal to the current flowing out of v.
- Ohm's law: potential difference is equal to current times resistance.

Resistance is a metric

Theorem

The function d, where $d(v, w)$ is the effective resistance between vertices v and w, is a metric on Ω.

We call this metric the resistance distance.

Resistance is a metric

Theorem

The function d, where $d(v, w)$ is the effective resistance between vertices v and w, is a metric on Ω.

We call this metric the resistance distance.
The theorem remains true if we replace the 1-ohm resistors by resistors of any given positive resistance.

Resistance is a metric

Theorem

The function d, where $d(v, w)$ is the effective resistance between vertices v and w, is a metric on Ω.

We call this metric the resistance distance.
The theorem remains true if we replace the 1-ohm resistors by resistors of any given positive resistance.
Now small average resistance distance between nodes indicates a good network. This notion is made precise in statistics by the notion of A-optimality for a block design: an A-optimal design, used for comparing a number of treatments, minimizes the average variance of the estimators of treatment differences.

Resistance is a metric

Theorem

The function d, where $d(v, w)$ is the effective resistance between vertices v and w, is a metric on Ω.

We call this metric the resistance distance.
The theorem remains true if we replace the 1-ohm resistors by resistors of any given positive resistance.
Now small average resistance distance between nodes indicates a good network. This notion is made precise in statistics by the notion of A-optimality for a block design: an A-optimal design, used for comparing a number of treatments, minimizes the average variance of the estimators of treatment differences.
Thus in our two example networks, the average resistance in the Petersen graph is $11/15$; that in the other network is $206/135$, more than twice as high.

Computing resistance distance

There are several methods for computing resistance distance.

Computing resistance distance

There are several methods for computing resistance distance.

- By hand, using the physical laws, for networks which are not too large.

Computing resistance distance

There are several methods for computing resistance distance.

- By hand, using the physical laws, for networks which are not too large.
- By matrix inversion. We weight each edge by its conductance, the reciprocal of the resistance (so that non-adjacent pairs of vertices have conductance zero). Then form the Laplacian matrix of the weighted network, and compute its Moore–Penrose inverse M. The effective resistance between v and w is $M_{vv} + M_{ww} - M_{vw} - M_{wv}$.

Computing resistance distance

There are several methods for computing resistance distance.

- By hand, using the physical laws, for networks which are not too large.
- By matrix inversion. We weight each edge by its conductance, the reciprocal of the resistance (so that non-adjacent pairs of vertices have conductance zero). Then form the Laplacian matrix of the weighted network, and compute its Moore–Penrose inverse M. The effective resistance between v and w is $M_{vv} + M_{ww} - M_{vw} - M_{wv}$.
- With weights as before, the effective resistance between v and w is equal to the sum of the weights of 2-component spanning forests with v and w in different components, divided by the sum of the weights of the spanning trees.

Resistance distance transform

The method proposed by Kagan and Klin works as follows.

Resistance distance transform

The method proposed by Kagan and Klin works as follows.

- Starting from a partition Π, assign weights to the edges in each part of Π. (For example, if we begin with a graph, assign 1 to the edges and 0 to the non-edges.)

Resistance distance transform

The method proposed by Kagan and Klin works as follows.

- Starting from a partition Π, assign weights to the edges in each part of Π. (For example, if we begin with a graph, assign 1 to the edges and 0 to the non-edges.)
- Compute the matrix of effective resistances.

Resistance distance transform

The method proposed by Kagan and Klin works as follows.

- Starting from a partition Π, assign weights to the edges in each part of Π. (For example, if we begin with a graph, assign 1 to the edges and 0 to the non-edges.)
- Compute the matrix of effective resistances.
- Now the new partition has two pairs in the same part if their effective resistances are equal.

Resistance distance transform

The method proposed by Kagan and Klin works as follows.

- Starting from a partition Π, assign weights to the edges in each part of Π. (For example, if we begin with a graph, assign 1 to the edges and 0 to the non-edges.)
- Compute the matrix of effective resistances.
- Now the new partition has two pairs in the same part if their effective resistances are equal.
- Iterate until the partition stabilises. (Typically, at the next stage, use as weights the conductances found in the previous step.)

Resistance distance transform

The method proposed by Kagan and Klin works as follows.

- Starting from a partition Π, assign weights to the edges in each part of Π. (For example, if we begin with a graph, assign 1 to the edges and 0 to the non-edges.)
- Compute the matrix of effective resistances.
- Now the new partition has two pairs in the same part if their effective resistances are equal.
- Iterate until the partition stabilises. (Typically, at the next stage, use as weights the conductances found in the previous step.)

An advantage of this method is that it "sees" the whole graph at each step, unlike the WL algorithm which only looks locally. In many cases they found that the partition was stable after just one step.

Drawbacks

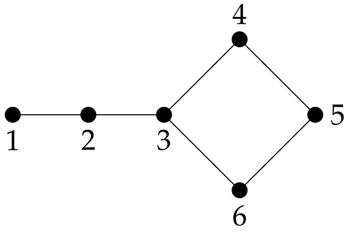
However, the method has some drawbacks.

Drawbacks

However, the method has some drawbacks.
While in WL each partition is guaranteed to refine the preceding one, that is not true for RDT.

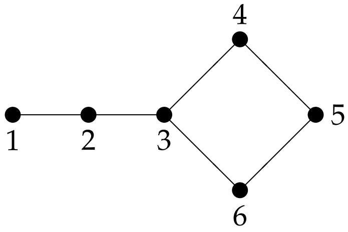
Drawbacks

However, the method has some drawbacks.
While in WL each partition is guaranteed to refine the preceding one, that is not true for RDT.



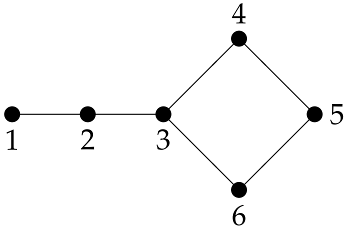
Drawbacks

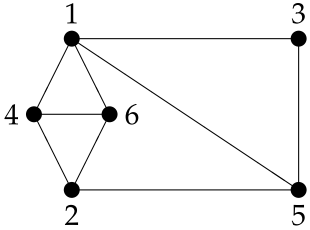
However, the method has some drawbacks.
While in WL each partition is guaranteed to refine the preceding one, that is not true for RDT.



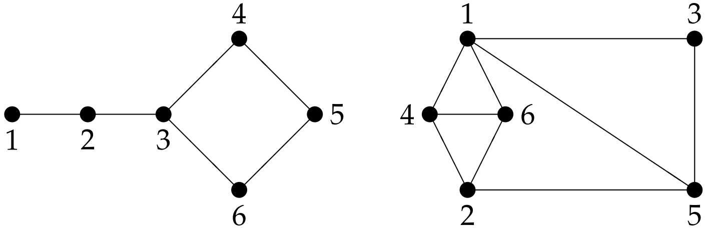
$\{2, 3\}$ is an edge; $\{4, 6\}$ is a non-edge; but $R_{23} = R_{46} = 1$.
In this case, the partition defined by resistance distance does not refine the original partition

Drawbacks, 2



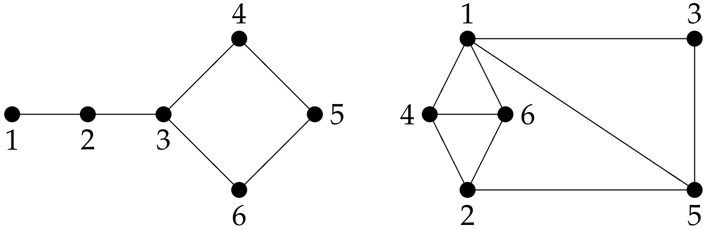


Drawbacks, 2



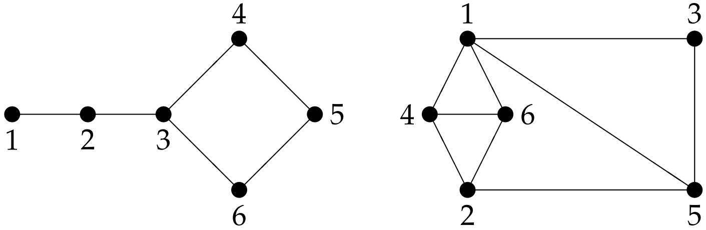
In the complementary graph Γ', for resistance distances not involving 4 or 6, we can replace the left-hand side by a single edge between 1 and 2.

Drawbacks, 2



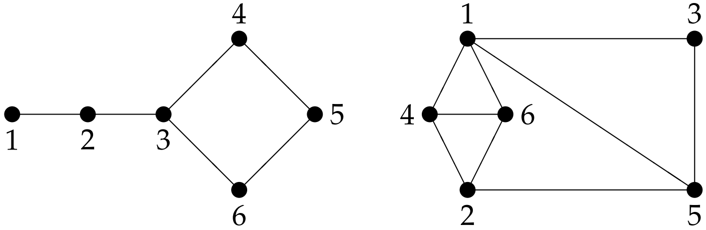
In the complementary graph Γ', for resistance distances not involving 4 or 6, we can replace the left-hand side by a single edge between 1 and 2. Therefore $R'_{12} = R'_{13}$.

Drawbacks, 2



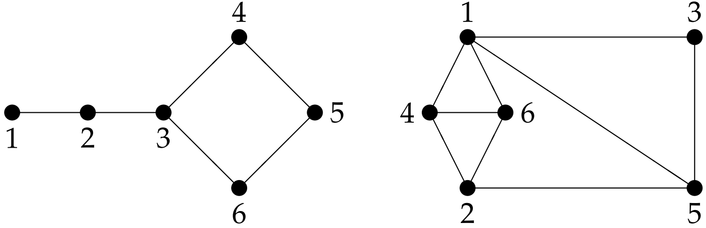
In the complementary graph Γ', for resistance distances not involving 4 or 6, we can replace the left-hand side by a single edge between 1 and 2. Therefore $R'_{12} = R'_{13}$. But $R_{12} \neq R_{13}$.

Drawbacks, 2



In the complementary graph Γ', for resistance distances not involving 4 or 6, we can replace the left-hand side by a single edge between 1 and 2. Therefore $R'_{12} = R'_{13}$. But $R_{12} \neq R_{13}$. Neither of these RDT partitions refines the other.

Drawbacks, 2



In the complementary graph Γ', for resistance distances not involving 4 or 6, we can replace the left-hand side by a single edge between 1 and 2. Therefore $R'_{12} = R'_{13}$. But $R_{12} \neq R_{13}$. Neither of these RDT partitions refines the other.
But we would like a graph and its complement to reduce to the same configuration since they are the same partition of Ω^2.

RDT2

In an attempt to avoid these problems, Mike Kagan proposed a new version RDT2.

RDT2

In an attempt to avoid these problems, Mike Kagan proposed a new version RDT2.
In this, starting from a partition, we associate independent indeterminates with the parts of the partition, and the initial matrix has these indeterminates in the corresponding positions.

RDT2

In an attempt to avoid these problems, Mike Kagan proposed a new version RDT2.
In this, starting from a partition, we associate independent indeterminates with the parts of the partition, and the initial matrix has these indeterminates in the corresponding positions.
Now when we compute the matrix of effective resistances, each entry is a rational function in these indeterminates.

RDT2

In an attempt to avoid these problems, Mike Kagan proposed a new version RDT2.
In this, starting from a partition, we associate independent indeterminates with the parts of the partition, and the initial matrix has these indeterminates in the corresponding positions.
Now when we compute the matrix of effective resistances, each entry is a rational function in these indeterminates.
RDT2 resolves both the earlier problems with RDT I showed you. I think that Mike Kagan will give more detail on this.

RDT2

In an attempt to avoid these problems, Mike Kagan proposed a new version RDT2.
In this, starting from a partition, we associate independent indeterminates with the parts of the partition, and the initial matrix has these indeterminates in the corresponding positions.
Now when we compute the matrix of effective resistances, each entry is a rational function in these indeterminates.
RDT2 resolves both the earlier problems with RDT I showed you. I think that Mike Kagan will give more detail on this.

Conjecture

Using RDT2, the output partition of each stage refines the input partition; hence the procedure is guaranteed to terminate.

Another feature

The matrix of resistance distances is necessarily symmetric. So, if the WL algorithm produces a coherent configuration whose matrices are not all symmetric, RDT2 cannot reach this configuration.

Another feature

The matrix of resistance distances is necessarily symmetric. So, if the WL algorithm produces a coherent configuration whose matrices are not all symmetric, RDT2 cannot reach this configuration.
There is an operation of Jordan product defined on symmetric matrices: $A * B = \frac{1}{2}(AB + BA)$. One can give axioms for this operation, and define a Jordan algebra to be an algebra in which these axioms hold.

Another feature

The matrix of resistance distances is necessarily symmetric. So, if the WL algorithm produces a coherent configuration whose matrices are not all symmetric, RDT2 cannot reach this configuration.

There is an operation of Jordan product defined on symmetric matrices: $A * B = \frac{1}{2}(AB + BA)$. One can give axioms for this operation, and define a Jordan algebra to be an algebra in which these axioms hold.

Thus (and for us this is the important example) the set of real symmetric matrices, with the operation $A * B = \frac{1}{2}(AB + BA)$, is a Jordan algebra.

Jordan schemes

We define a Jordan scheme to be an object satisfying similar axioms to those of a coherent configuration, a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying:

- Every ordered pair satisfies exactly one of the relations.

Jordan schemes

We define a Jordan scheme to be an object satisfying similar axioms to those of a coherent configuration, a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying:

- Every ordered pair satisfies exactly one of the relations.
- The relation of equality (the diagonal of $\Omega \times \Omega$) is the one of these relations.

Jordan schemes

We define a Jordan scheme to be an object satisfying similar axioms to those of a coherent configuration, a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying:

- Every ordered pair satisfies exactly one of the relations.
- The relation of equality (the diagonal of $\Omega \times \Omega$) is the one of these relations.
- All the relations are symmetric.

Jordan schemes

We define a Jordan scheme to be an object satisfying similar axioms to those of a coherent configuration, a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying:

- Every ordered pair satisfies exactly one of the relations.
- The relation of equality (the diagonal of $\Omega \times \Omega$) is the one of these relations.
- All the relations are symmetric.
- Given $i, j, k \in \{1, \ldots, r\}$ and $(x, y) \in R_i$, the number of $z \in \Omega$ such that $(x, z) \in R_j$ and $(z, y) \in R_k$ plus the number such that $(y, z) \in R_j$ and $(z, x) \in R_k$ depends only on (i, j, k) and not on the choice of x and y.

Jordan schemes

We define a Jordan scheme to be an object satisfying similar axioms to those of a coherent configuration, a collection of binary relations $R_1, \ldots, R_r$ on a set Ω satifying:

- Every ordered pair satisfies exactly one of the relations.
- The relation of equality (the diagonal of $\Omega \times \Omega$) is the one of these relations.
- All the relations are symmetric.
- Given $i, j, k \in \{1, \ldots, r\}$ and $(x, y) \in R_i$, the number of $z \in \Omega$ such that $(x, z) \in R_j$ and $(z, y) \in R_k$ plus the number such that $(y, z) \in R_j$ and $(z, x) \in R_k$ depends only on (i, j, k) and not on the choice of x and y.

The span over the real numbers of the matrices of a Jordan scheme is thus a Jordan algebra. One can define a Jordan WL algorithm which will produce the largest Jordan scheme below a given partition.

Jordan schemes, 2

If we take a homogeneous coherent configuration, and symmetrise the matrices by adding any non-symmetric matrix and its transpose, we obtain a Jordan scheme.

Jordan schemes, 2

If we take a homogeneous coherent configuration, and symmetrise the matrices by adding any non-symmetric matrix and its transpose, we obtain a Jordan scheme.
A Jordan scheme is called proper if it is not obtained in this manner.

Jordan schemes, 2

If we take a homogeneous coherent configuration, and symmetrise the matrices by adding any non-symmetric matrix and its transpose, we obtain a Jordan scheme.

A Jordan scheme is called proper if it is not obtained in this manner.

Mikhail Muzychuk, Sven Reichard, and Mikhail Klin have produced infinite families of examples of proper Jordan schemes.

Two conjectures

Conjecture

A configuration is stable under RDT2 if and only if it is a Jordan scheme.

Two conjectures

Conjecture

A configuration is stable under RDT2 if and only if it is a Jordan scheme.

Conjecture

The stable configuration obtained from a given partition Π *is the same as the Jordan scheme obtained by applying the Jordan WL algorithm to* Π*.*

Two conjectures

Conjecture

A configuration is stable under RDT2 if and only if it is a Jordan scheme.

Conjecture

The stable configuration obtained from a given partition Π *is the same as the Jordan scheme obtained by applying the Jordan WL algorithm to* Π.

And we hope that RDT2 will stabilise in significantly fewer steps than the Jordan WL algorithm! We also hope that new examples of Jordan schemes can be produced in this way.

Two conjectures

Conjecture

A configuration is stable under RDT2 if and only if it is a Jordan scheme.

Conjecture

The stable configuration obtained from a given partition Π *is the same as the Jordan scheme obtained by applying the Jordan WL algorithm to* Π.

And we hope that RDT2 will stabilise in significantly fewer steps than the Jordan WL algorithm! We also hope that new examples of Jordan schemes can be produced in this way. We have made some progress towards these conjectures but they are not yet established.

… for your attention.