Giant Components in Random Temporal Graphs

Michael Raskin

Joint works with Ruben Becker, Arnaud Casteigts, Pierluigi Crescenzi, Bojana Kodric, Malte Renken, Viktor Zamaraev,

LaBRI, University of Bordeaux

September 2023

RANDOM 2023

Temporal graphs: context

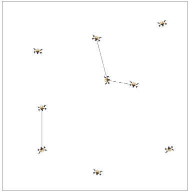
Consider use of a UAV for aerial photography...

Temporal graphs: context

Consider use of UAV**s** for aerial photography...

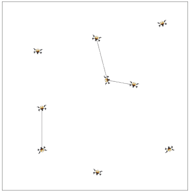
Temporal graphs: context

Consider use of UAV**s** for aerial photography...



... The flock does not always have good connectivity!

Consider use of UAV**s** for aerial photography...

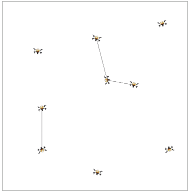


... The flock does not always have good connectivity!

... but the data can be relayed

Temporal graphs: context

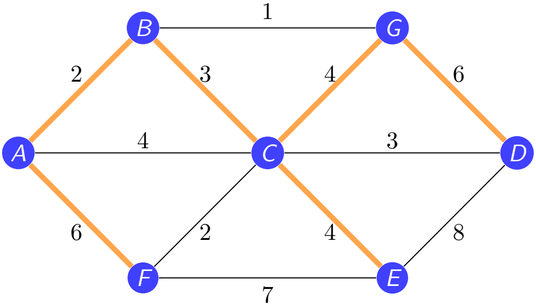
Consider use of UAV**s** for aerial photography...



... The flock does not always have good connectivity!
... but the data can be relayed

With UAVs as nodes, add temporary edges when connections are useable
We study sequences of edge-E-at-time-T for relayed data to take

Temporal graphs and paths

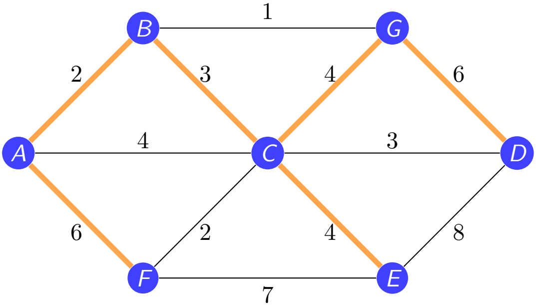


Definition: Temporal graph

Temporal graph is a graph with edge presence times

Temporal path: path with edges crossed at increasing presence times

Temporal graphs and paths

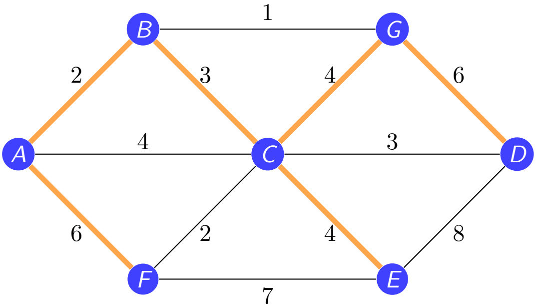


Temporal graph: graph with edge presence times

Definition: Temporal path

Temporal path is a path with edges crossed at increasing presence times

Temporal graphs and paths

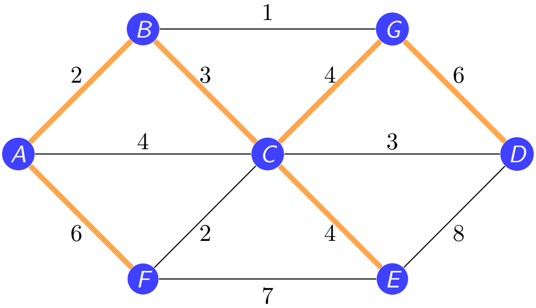


Temporal graph: graph with edge presence times

Temporal path: path with edges crossed at increasing presence times

$A - B - C - G - D$: temporal path, times: $2 < 3 < 4 < 6$
$A - C - D$: not temporal path, times: $4 > 3$

Temporal graphs and paths



Temporal path: path with edges crossed at increasing presence times

$A - B - C - G - D$: temporal path, times: $2 < 3 < 4 < 6$
$A - C - D$: not temporal path, times: $4 > 3$

$A - C - E - D$: definition-dependent! times: $4 = 4 < 8$

Temporal graph reachability

$u \rightsquigarrow v$: there is a temporal path from u to v

- Unlike undirected graphs, $\rightsquigarrow$ is **not symmetric**
  $A \overset{1}{—} B \overset{2}{—} C$: $A \rightsquigarrow C$, but $C \not\rightsquigarrow A$
- Unlike static (non-temporal) graphs, $\rightsquigarrow$ is **not transitive**
  $C \rightsquigarrow B$, $B \rightsquigarrow A$, but $C \not\rightsquigarrow A$
- Many obvious facts about static graph do not translate
  If we have a source, a sink, and a path from the sink to the source…
  still **not always temporally connected**!
- A journey with multiple changes

Temporal graph reachability

$u \rightsquigarrow v$: there is a temporal path from u to v

- Unlike undirected graphs, $\rightsquigarrow$ is **not symmetric**
  $A \overset{1}{—} B \overset{2}{—} C$: $A \rightsquigarrow C$, but $C \not\rightsquigarrow A$
- Unlike static (non-temporal) graphs, $\rightsquigarrow$ is **not transitive**
  $C \rightsquigarrow B$, $B \rightsquigarrow A$, but $C \not\rightsquigarrow A$
- Many obvious facts about static graph do not translate
  If we have a source, a sink, and a path from the sink to the source...
  still **not always temporally connected**!
- A journey with multiple changes

Temporal graph reachability

$u \rightsquigarrow v$: there is a temporal path from u to v

- Unlike undirected graphs, $\rightsquigarrow$ is **not symmetric**
  $A \overset{1}{—} B \overset{2}{—} C$: $A \rightsquigarrow C$, but $C \not\rightsquigarrow A$
- Unlike static (non-temporal) graphs, $\rightsquigarrow$ is **not transitive**
  $C \rightsquigarrow B$, $B \rightsquigarrow A$, but $C \not\rightsquigarrow A$
- Many obvious facts about static graph do not translate
  If we have a source, a sink, and a path from the sink to the source...
  still **not always temporally connected**!
- A journey with multiple changes

Temporal graph reachability

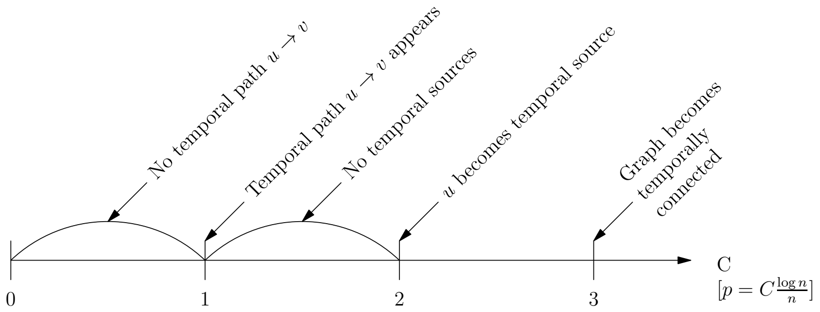
$u \rightsquigarrow v$: there is a temporal path from u to v

- Unlike undirected graphs, $\rightsquigarrow$ is **not symmetric**
  $A \overset{1}{—} B \overset{2}{—} C$: $A \rightsquigarrow C$, but $C \not\rightsquigarrow A$
- Unlike static (non-temporal) graphs, $\rightsquigarrow$ is **not transitive**
  $C \rightsquigarrow B$, $B \rightsquigarrow A$, but $C \not\rightsquigarrow A$
- Many obvious facts about static graph do not translate
  If we have a source, a sink, and a path from the sink to the source...
  still **not always temporally connected**!
- A journey with multiple changes

Our focus

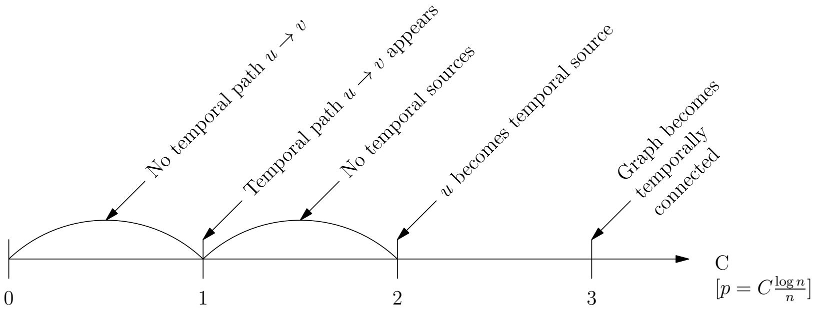
Definition: RSTG $\mathcal{F}_{n,p}$

Random Simple Temporal Graph (RSTG) is an Erdős-Rényi random graph with uniformly random (strict) edge order
Notation: $\mathcal{F}_{n,p}$



Our focus

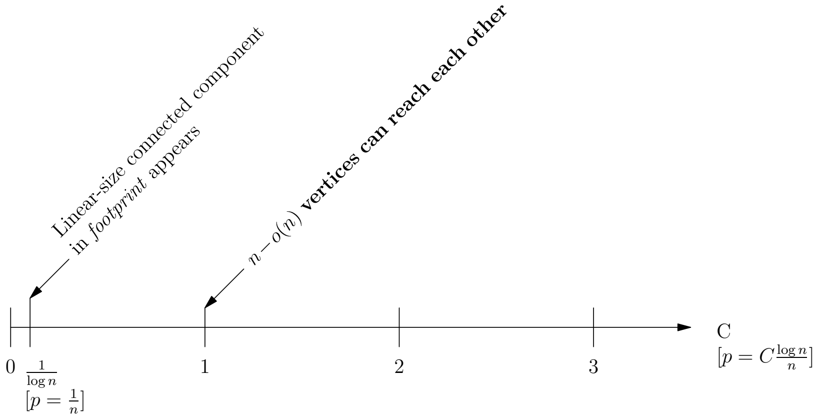
RSTG $\mathcal{F}_{n,p}$: Erdős-Réniy random graph with random edge order



We study temporal connected components

Our focus

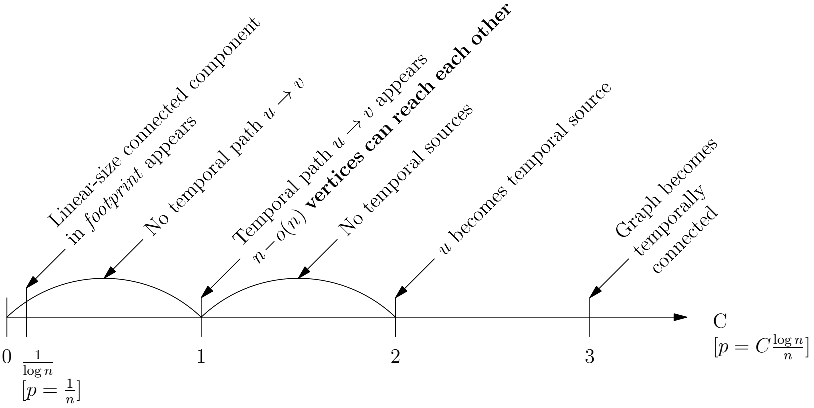
RSTG $\mathcal{F}_{n,p}$: Erdős-Rényi random graph with random edge order



We study temporal connected components

Our focus

RSTG $\mathcal{F}_{n,p}$: Erdős-Réniy random graph with random edge order



We study temporal connected components

Connected components

What is a (strongly) connected component?
«Everyone reaches everyone»

Inside or outside the component?
Normally irrelevant: A reaches B via C, then C reaches B and *everyone reached by B*

No transitivity for temporal reachability!

- *Open* connected components:
  every node in the component reaches every other node
  ... using paths that *may* leave the component and re-enter it
- *Closed* connected components:
  every node in the component reaches every other node
  via paths *inside* the component

In our random setting both happens roughly at the same time
Proofs simpler for the open case

Connected components

What is a (strongly) connected component?
«Everyone reaches everyone»

Inside or outside the component?
Normally irrelevant: A reaches B via C, then C reaches B and *everyone reached by B*

No transitivity for temporal reachability!

- *Open* connected components:
  every node in the component reaches every other node
  ... using paths that *may* leave the component and re-enter it
- *Closed* connected components:
  every node in the component reaches every other node
  via paths *inside* the component

In our random setting both happens roughly at the same time
Proofs simpler for the open case

Connected components

What is a (strongly) connected component?
«Everyone reaches everyone»

Inside or outside the component?
Normally irrelevant: A reaches B via C, then C reaches B
and *everyone reached by B*

No transitivity for temporal reachability!

- *Open* connected components:
  every node in the component reaches every other node
  … using paths that *may* leave the component and re-enter it
- *Closed* connected components:
  every node in the component reaches every other node
  via paths *inside* the component

In our random setting both happens roughly at the same time
Proofs simpler for the open case

Connected components

What is a (strongly) connected component?
«Everyone reaches everyone»

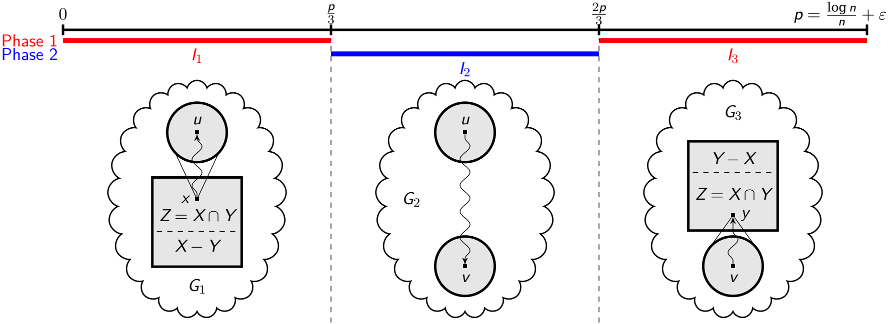
Inside or outside the component?
Normally irrelevant: A reaches B via C, then C reaches B
and *everyone reached by* B

No transitivity for temporal reachability!

- *Open* connected components:
  every node in the component reaches every other node
  … using paths that *may* leave the component and re-enter it
- *Closed* connected components:
  every node in the component reaches every other node
  via paths *inside* the component

In our random setting both happens roughly at the same time
Proofs simpler for the open case

Construction: Open component

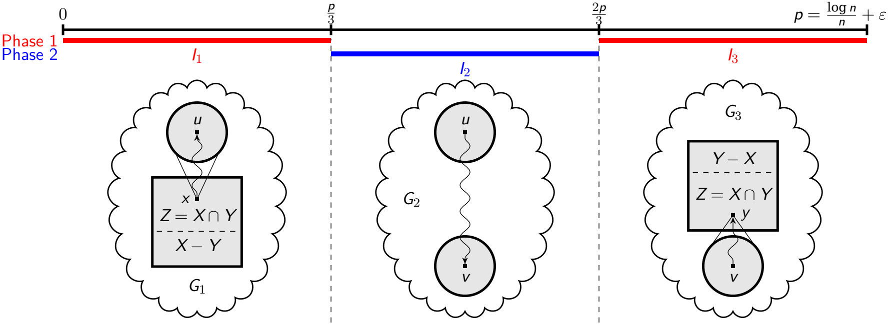


Construction: Open component

First phase: a typical vertex reaches $\omega(n^{1/3})$ vertices

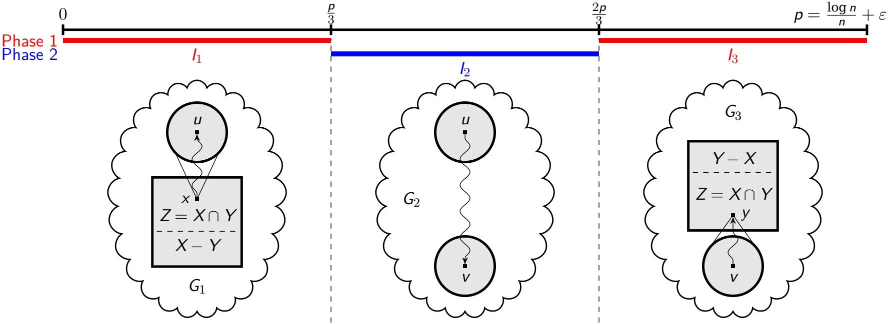
Last phase: a typical vertex is reached by $\omega(n^{1/3})$ vertices

Middle phase: any two large sets of vertices are connected —
despite loss of the edges used for first/last phase



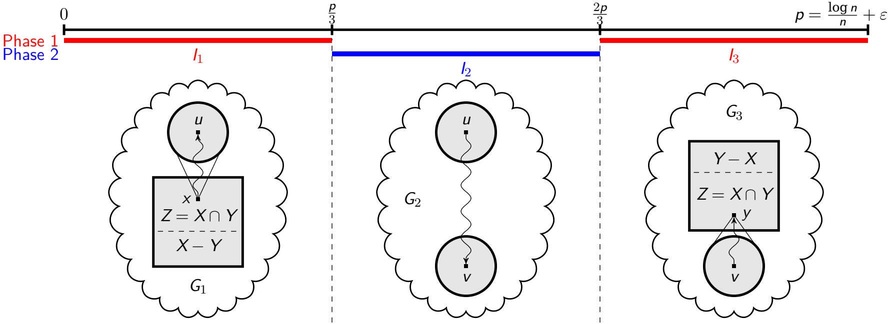
Construction: Open component

First phase: a typical vertex reaches $\omega(n^{1/3})$ vertices
Last phase: a typical vertex is reached by $\omega(n^{1/3})$ vertices
Middle phase: any two large sets of vertices are connected —
despite loss of the edges used for first/last phase



Construction: Open component

First phase: a typical vertex reaches $\omega(n^{1/3})$ vertices
Last phase: a typical vertex is reached by $\omega(n^{1/3})$ vertices
Middle phase: any two large sets of vertices are connected —
despite loss of the edges used for first/last phase



Choice of the vertices

- Normally takes $\frac{\log n}{n}$ time and $O(\log n)$ hops for $u \rightsquigarrow v$
- But a vertex can «sleep» with no edges for some time
- Longest observed sleeping time: $\frac{\log n}{n}$
- Sleep determines extra $\frac{\log n}{n}$ per level of generality!
  typical pair $\rightarrow$ source/sink $\rightarrow$ full connectivity
- Most vertices are not sleepers, yield a closed connected component

Choice of the vertices

- Normally takes $\frac{\log n}{n}$ time and $O(\log n)$ hops for $u \rightsquigarrow v$
- But a vertex can «sleep» with no edges for some time
- Longest observed sleeping time: $\frac{\log n}{n}$
- Sleep determines extra $\frac{\log n}{n}$ per level of generality!
  typical pair $\to$ source/sink $\to$ full connectivity
- Most vertices are not sleepers, yield a closed connected component

Choice of the vertices

- Normally takes $\frac{\log n}{n}$ time and $O(\log n)$ hops for $u \rightsquigarrow v$
- But a vertex can «sleep» with no edges for some time
- Longest observed sleeping time: $\frac{\log n}{n}$
- Sleep determines extra $\frac{\log n}{n}$ per level of generality!
  typical pair $\rightarrow$ source/sink $\rightarrow$ full connectivity
- Most vertices are not sleepers, yield a closed connected component

Choice of the vertices

- Normally takes $\frac{\log n}{n}$ time and $O(\log n)$ hops for $u \rightsquigarrow v$
- But a vertex can «sleep» with no edges for some time
- Longest observed sleeping time: $\frac{\log n}{n}$
- Sleep determines extra $\frac{\log n}{n}$ per level of generality!
  typical pair $\rightarrow$ source/sink $\rightarrow$ full connectivity
- Most vertices are not sleepers, yield a closed connected component

Construction: Foremost tree

Building block for larger construction

Tree of temporal paths

- Start with a single vertex

    advanced version: and some starting time

- Add earliest later edge that adds a new vertex to tree
- Repeat

Same can be done in reverse

RSTG — Reachability-Equivalent Definitions

- $\mathcal{G}_{n,p}$ (Erdős–Rényi random graph) with random edge order
- $\mathcal{G}_{n,p}$ with random edge labels from $[0;1]$
- $\mathcal{G}_{n,p}$ with random edge labels from $[0;p]$
- K_n (clique) random edge labels from $[0;1]$, edges after p removed

When things happen as p grows for fixed labelling of K_n?

RSTG — Reachability-Equivalent Definitions

- $\mathcal{G}_{n,p}$ (Erdős–Rényi random graph) with random edge order
- $\mathcal{G}_{n,p}$ with random edge labels from $[0; 1]$
- $\mathcal{G}_{n,p}$ with random edge labels from $[0; p]$
- K_n (clique) random edge labels from $[0; 1]$, edges after p removed

When things happen as p grows for fixed labelling of K_n?

RSTG — Reachability-Equivalent Definitions

- $\mathcal{G}_{n,p}$ (Erdős–Rényi random graph) with random edge order
- $\mathcal{G}_{n,p}$ with random edge labels from $[0;1]$
- $\mathcal{G}_{n,p}$ with random edge labels from $[0;p]$
- K_n (clique) random edge labels from $[0;1]$, edges after p removed

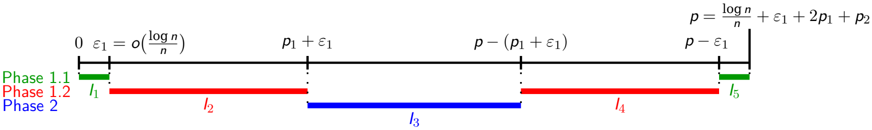
When things happen as p grows for fixed labelling of K_n?

RSTG — Reachability-Equivalent Definitions

- $\mathcal{G}_{n,p}$ (Erdős–Rényi random graph) with random edge order
- $\mathcal{G}_{n,p}$ with random edge labels from $[0;1]$
- $\mathcal{G}_{n,p}$ with random edge labels from $[0;p]$
- K_n (clique) random edge labels from $[0;1]$, edges after p removed

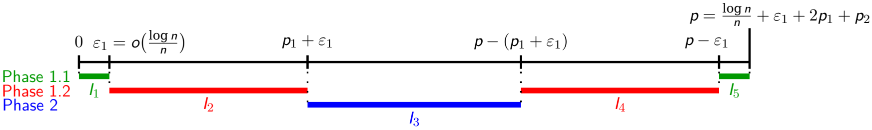
When things happen as p grows for fixed labelling of K_n?

Closed components



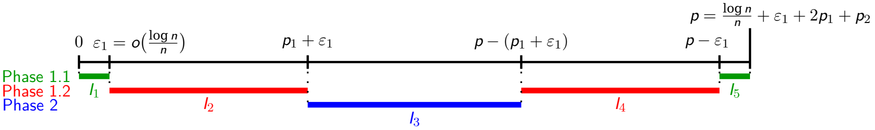
Closed components

- Similar to the open case
- Extra sub-phases added in the beginning/in the end
  - Chosen vertices quickly reach $(\log n)^{\Theta(1)}$ vertices without going through the excluded ones
  - Concentration of growth speed afterwards — starting set large enough



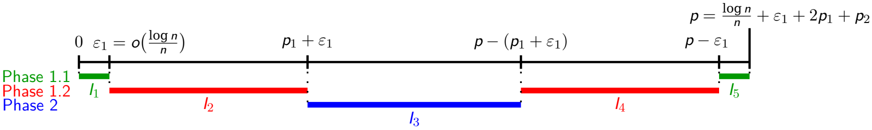
Closed components

- Similar to the open case
- Extra sub-phases added in the beginning/in the end
  - Chosen vertices quickly reach $(\log n)^{\Theta(1)}$ vertices without going through the excluded ones
  - Concentration of growth speed afterwards — starting set large enough



Closed components

- Similar to the open case
- Extra sub-phases added in the beginning/in the end
  - Chosen vertices quickly reach $(\log n)^{\Theta(1)}$ vertices without going through the excluded ones
  - Concentration of growth speed afterwards — starting set large enough



Conclusion

- Giant connected component arises once vertices are «typically» connected
- Sharp threshold between $o(n)$ and $n - o(n)$ size of connected component
- Techniques developed for multi-phase analysis of temporal reachability

Conclusion

Thanks for your attention!

Questions?

Conclusion

- Giant connected component arises once vertices are «typically» connected
- Sharp threshold between $o(n)$ and $n - o(n)$ size of connected component
- Techniques developed for multi-phase analysis of temporal reachability