The Peculiar Phase Structure of Random Graph Bisection

Allon G. Percus

School of Mathematical Sciences
Claremont Graduate University

April 8, 2011

Collaborators

- Gabriel Istrate
- Bruno Gonçalves
- Robert Sumi
- Stefan Boettcher

Journal of Mathematical Physics **49**, 125219 (2008).

Outline

1. Background
   - Random Combinatorial Optimization
   - Phase Structure
   - Clustering Transition

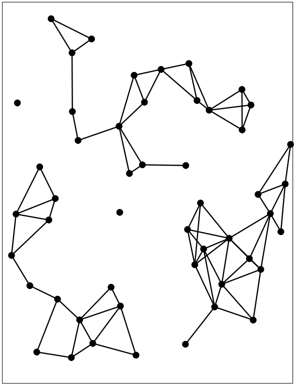
2. Graph Bisection
   - Definition and Previous Results
   - Upper Bound
   - Computational Consequences

Outline

1. Background
   - Random Combinatorial Optimization
   - Phase Structure
   - Clustering Transition

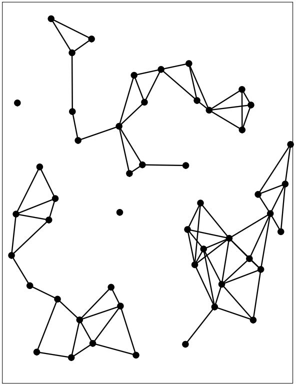
2. Graph Bisection
   - Definition and Previous Results
   - Upper Bound
   - Computational Consequences

Graph Coloring



- Applications: channel assignment, scheduling, etc.
- Graph $G = (V, E)$

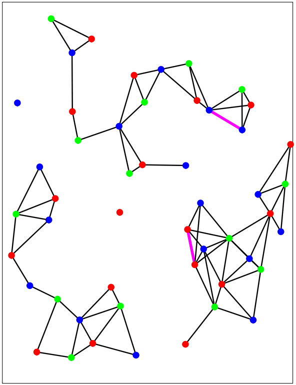
Graph Coloring



- Applications: channel assignment, scheduling, etc.

- Graph $G = (V, E)$
- At each vertex $v \in V$, assign color $c(v) \in \{1, \ldots, q\}$
- Minimize $|(u, v) \in E : c(u) = c(v)|$: number of edges connecting two vertices with same color

Graph Coloring



$q = 3$

- Applications: channel assignment, scheduling, etc.

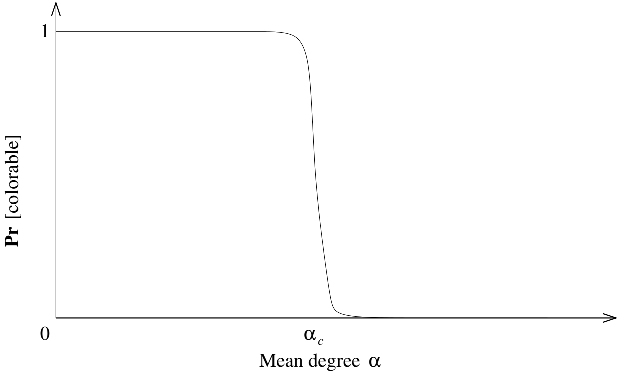
- Graph $G = (V, E)$
- At each vertex $v \in V$, assign color $c(v) \in \{1, \ldots, q\}$
- Minimize $|(u, v) \in E : c(u) = c(v)|$: number of edges connecting two vertices with same color

Algorithmic Complexity

- Worst-case complexity: NP-hard for $q \geq 3$.
- What about typical-case complexity over instances from a random generative model?
- Theoretical understanding starts with study of unstructured cases.
- Take Erdős-Rényi $\mathcal{G}_{np}$ model: n vertices ($|V| = n$), edges placed on pairs of vertices ($(u, v) \in E$) with fixed probability p, independently of one another.

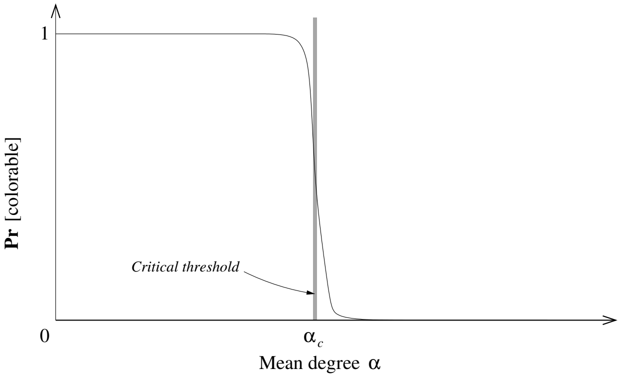
Phase Transition

Over $\mathcal{G}_{np}$ random graph ensemble, phase transition:



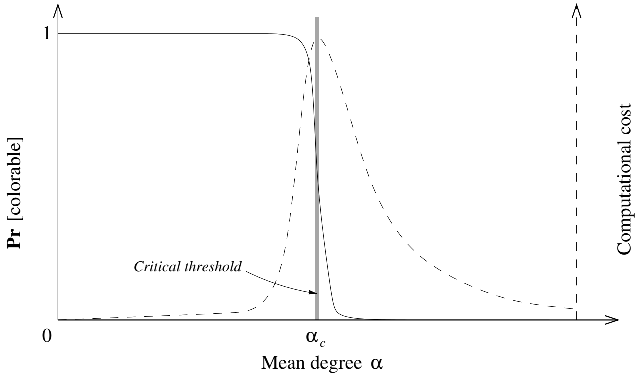
Phase Transition

Over $\mathcal{G}_{np}$ random graph ensemble, phase transition:



Phase Transition

Over $\mathcal{G}_{np}$ random graph ensemble, phase transition:



Also easy-hard-easy pattern: hard instances concentrated near phase boundary! [Cheeseman et al, 1991; Mitchell et al, 1992]

Connection to Complexity

Empirically:

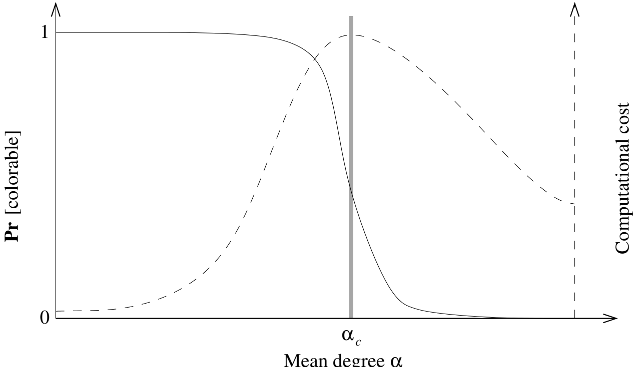
- Property holds for a wide range of algorithms.
- Connection between phase structure and typical-case algorithmic complexity is seen in numerous other random combinatorial problems as well: satisfiability, vertex cover, etc.

Detailed Phase Structure

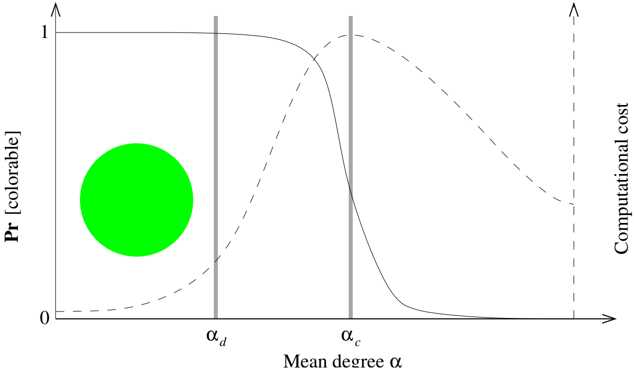
More can be learned by considering space of all optimal colorings of a graph.

- Define two solutions to be adjacent if Hamming distance is small: at most $o(n)$ variables differ in value.
- For small α, all solutions lie in a single “cluster”: any two solutions are linked by a path of adjacent solutions. *(Replica symmetric phase.)*

Detailed Phase Structure

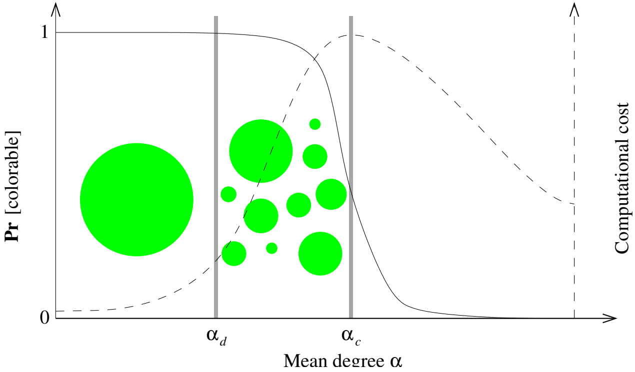


Detailed Phase Structure



Below a new threshold $\alpha_d < \alpha_c$: single solution cluster.

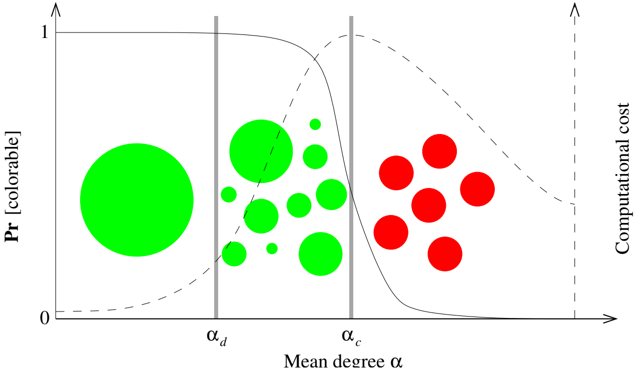
Detailed Phase Structure



Below a new threshold $\alpha_d < \alpha_c$: single solution cluster.

Above α_d: cluster fragments into multiple non-adjacent clusters.

Detailed Phase Structure



Below a new threshold $\alpha_d < \alpha_c$: single solution cluster.

Above α_d: cluster fragments into multiple non-adjacent clusters.

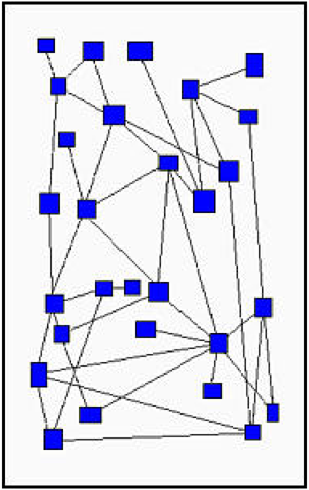
Algorithmic Consequences

- Cluster fragmentation is associated with formation of frozen variables: local backbone of variables that take on same value within a cluster of solutions.
- This traps algorithms: lots of colorings but hard to find them, making it a “hard colorable” subphase.
- But physical picture also motivates new algorithms: survey propagation explicitly takes account of cluster structure, fixing only those variables that are frozen within a cluster.

Outline

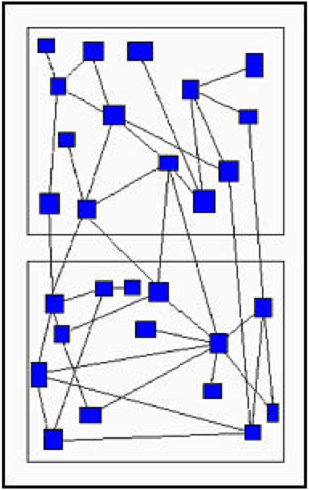
1. Background
   - Random Combinatorial Optimization
   - Phase Structure
   - Clustering Transition
2. Graph Bisection
   - Definition and Previous Results
   - Upper Bound
   - Computational Consequences

Definition



- Applications: computer chip design, resource allocation, image processing
- Graph $G = (V, E)$, $|V|$ even

Definition



- Applications: computer chip design, resource allocation, image processing
- Graph $G = (V, E)$, $|V|$ even
- Partition V into two disjoint subsets V_1 and V_2, $|V_1| = |V_2|$
- Minimize bisection width $w = |(u, v) \in E : u \in V_1, v \in V_2|$: number of edges with an endpoint in each subset

Worst-Case / Average-Case Complexity

- Corresponding decision problem is in P (solvable in polynomial time): is there a perfect bisection, i.e., $w = 0$?
- Optimization problem is NP-hard.
- What about over $\mathcal{G}_{np}$ ensemble?

Structure of $\mathcal{G}_{np}$ Graphs

Mean degree of graph is $\alpha = p(n-1)$. The following results on the birth of the giant component are known [Erdős-Rényi, 1959]:

- For $\alpha < 1$, only very small components exist: size $O(\log n)$.
- For $\alpha > 1$, there exists a giant component of expected size gn, $g = 1 - e^{-\alpha g}$. All other components: size $O(\log n)$.
- Expected fraction of isolated vertices is $(1-p)^{n-1} \approx e^{-\alpha}$.

Structure of $\mathcal{G}_{np}$ Graphs

Mean degree of graph is $\alpha = p(n-1)$. The following results on the birth of the giant component are known [Erdős-Rényi, 1959]:

- For $\alpha < 1$, only very small components exist: size $O(\log n)$.
- For $\alpha > 1$, there exists a giant component of expected size gn, $g = 1 - e^{-\alpha g}$. All other components: size $O(\log n)$.
  - At $\alpha = 2\log 2$, $g = 1/2$
- Expected fraction of isolated vertices is $(1-p)^{n-1} \approx e^{-\alpha}$.
  - At $\alpha = 2\log 2$, $n/4$ isolated vertices

Consequence: Bisection Width

Known results and bounds [Luczak & McDiarmid, 2001]:

- For $\alpha < 1$, $w = 0$ w.h.p.
  - Enough small components to guarantee perfect bisection

Consequence: Bisection Width

Known results and bounds [Luczak & McDiarmid, 2001]:

- For $\alpha < 1$, $w = 0$ w.h.p.
  - Enough small components to guarantee perfect bisection
- For $1 < \alpha < 2 \log 2$, also $w = 0$ w.h.p.
  - Even close to $\alpha = 2 \log 2$, where the giant component almost occupies entire partition, enough isolated vertices to guarantee perfect bisection

Consequence: Bisection Width

Known results and bounds [Luczak & McDiarmid, 2001]:

- For $\alpha < 1$, $w = 0$ w.h.p.
  - Enough small components to guarantee perfect bisection
- For $1 < \alpha < 2\log 2$, also $w = 0$ w.h.p.
  - Even close to $\alpha = 2\log 2$, where the giant component almost occupies entire partition, enough isolated vertices to guarantee perfect bisection
- For $\alpha > 2\log 2$, $w = \Omega(n)$ and obvious upper bound $w/n \leq \alpha/2$ w.h.p.

Consequence: Bisection Width

Known results and bounds [Luczak & McDiarmid, 2001]:

- For $\alpha < 1$, $w = 0$ w.h.p.
  - Enough small components to guarantee perfect bisection
- For $1 < \alpha < 2\log 2$, also $w = 0$ w.h.p.
  - Even close to $\alpha = 2\log 2$, where the giant component almost occupies entire partition, enough isolated vertices to guarantee perfect bisection
- For $\alpha > 2\log 2$, $w = \Omega(n)$ and obvious upper bound $w/n \leq \alpha/2$ w.h.p.
- For $2\log 2 < \alpha < 4\log 2$, $w/n \leq (\alpha - \log 2)/4$ w.h.p. [Goldberg & Lynch, 1985]

Consequence: Bisection Width

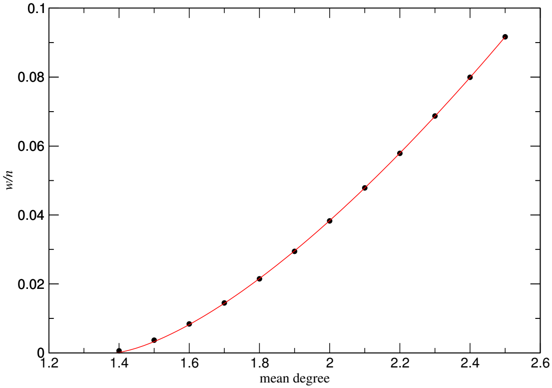
Known results and bounds [Luczak & McDiarmid, 2001]:

- For $\alpha < 1$, $w = 0$ w.h.p.
  - Enough small components to guarantee perfect bisection
- For $1 < \alpha < 2\log 2$, also $w = 0$ w.h.p.
  - Even close to $\alpha = 2\log 2$, where the giant component almost occupies entire partition, enough isolated vertices to guarantee perfect bisection
- For $\alpha > 2\log 2$, $w = \Omega(n)$ and obvious upper bound $w/n \leq \alpha/2$ w.h.p.
- For $2\log 2 < \alpha < 4\log 2$, $w/n \leq (\alpha - \log 2)/4$ w.h.p. [Goldberg & Lynch, 1985]

Still leaves a gap at $\alpha = 2\log 2$. Can we do better?

Consequence: Bisection Width

Experimental results [Boettcher & Percus, 1999]:

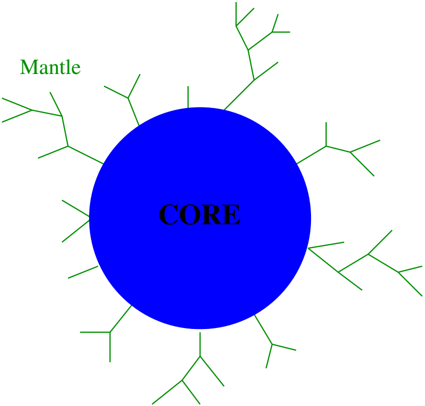


Consequence: Solution Structure

- For $\alpha < 2\log 2$, all solutions lie in a single cluster [Istrate, Kasiviswanathan & Percus, 2006]
  - Enough small components that any two solutions are connected by a chain of small swaps preserving balance constraint
- For $\alpha > 2\log 2$, solution space structure is determined by how giant component gets cut

Giant Component Structure

- Giant component consists of a mantle of trees and a remaining core [Pittel, 1990]
- Individual trees are of size $O(\log n)$
- But does optimal cut simply trim trees, or does it slice through core?



Cutting Trees

As long as core is smaller than $n/2$, we can at least get an upper bound on w by restricting cuts to trees.

Theorem

Let $\epsilon = \alpha - 2\log 2$. Then there exists an $\epsilon_0 > 0$ such that for every $\epsilon < \epsilon_0$, w.h.p.

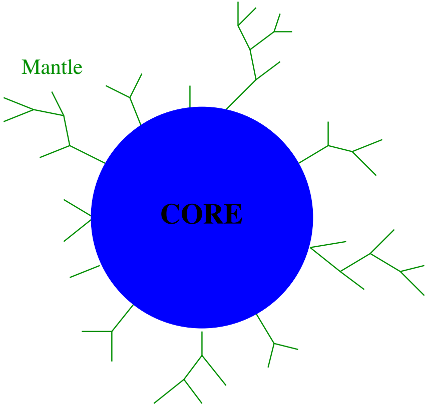
$$\frac{w}{n} < \frac{\epsilon}{\log 1/\epsilon}$$

for graphs with mean degree α in $\mathcal{G}_{np}$.

Among other things, this closes the gap at $\alpha = 2\log 2$.
Now how do we prove it?

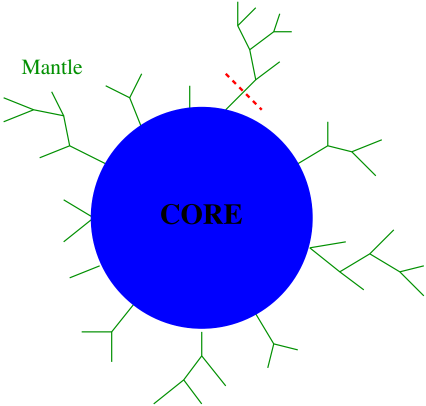
Cutting Trees

Cut trees starting from largest one until giant component is pruned to size $n/2$:



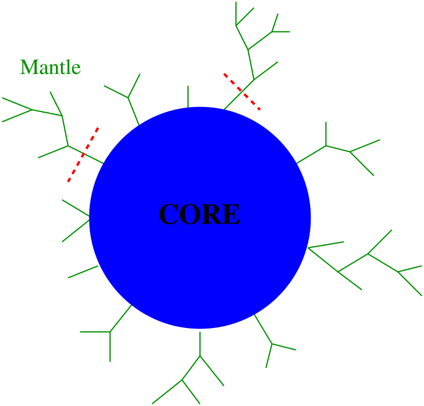
Cutting Trees

Cut trees starting from largest one until giant component is pruned to size $n/2$:



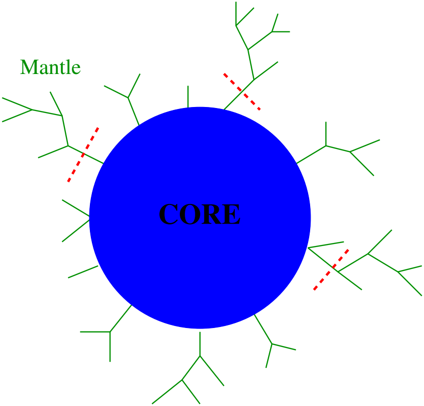
Cutting Trees

Cut trees starting from largest one until giant component is pruned to size $n/2$:



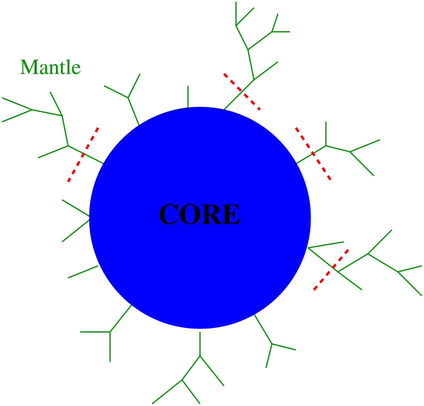
Cutting Trees

Cut trees starting from largest one until giant component is pruned to size $n/2$:



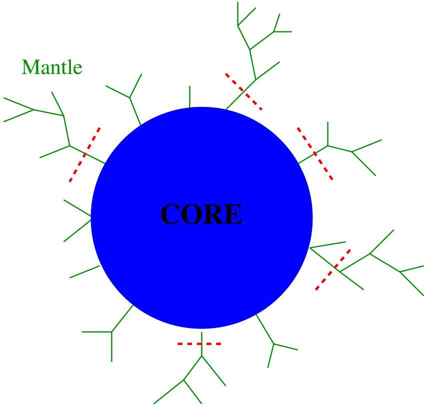
Cutting Trees

Cut trees starting from largest one until giant component is pruned to size $n/2$:



Cutting Trees

Cut trees starting from largest one until giant component is pruned to size $n/2$:



How Many Trees is Enough?

- Let δn be "excess" of giant component, $\delta = g - 1/2$.
  Let bn be number of nodes in mantle.
- Then δ/b is fraction of mantle's nodes to cut.
- Now find largest t_0 such that δ/b equals fraction of nodes living on trees of size $\geq t_0$.
- If $P(t)$ is distribution of tree sizes on mantle,

$$\frac{\delta}{b} = \frac{\sum_{t=t_0}^{\infty} tP(t)}{\sum_{t=1}^{\infty} tP(t)}$$

- The number of trees of size $\geq t_0$ is then

$$w' = \sum_{t=t_0}^{\infty} P(t) \frac{bn}{\sum_{t=1}^{\infty} tP(t)}$$

Distribution of Tree Sizes

Fortunate result of probabilistic independence in $\mathcal{G}_{np}$ [Janson et al, 2000]:

- $P(t)$ is simply given by # of ways of constructing tree of size t from q roots ($q = (g-b)n$, size of core) and r other nodes ($r = bn$, size of mantle).
- This is "just combinatorics":

$$P(t) = \binom{r}{t} t^t \frac{q}{r} \frac{(q+r-t)^{r-t+1}}{(q+r)^{r-1}}$$

- Let $\rho = b/g$. Then at large n,

$$P(t) \approx \frac{t^t e^{-\rho t}}{t!} \rho^{t-1}(1-\rho)$$

Upper Bound on Bisection Width

- We now have enough to calculate (or at least bound) w'. The rest of the proof is just cleaning up.
- That gives the upper bound we need on bisection width w.
- Theorem implies that w/n scales superlinearly in $\epsilon = \alpha - 2\log 2$ for small ϵ. This turns out to have physical and algorithmic consequences.
- This holds for every $\epsilon < \epsilon_0$, but ϵ_0 could be very small!

Expander Core of Giant Component

Look more closely at giant component structure. Define notion of expander graphs:

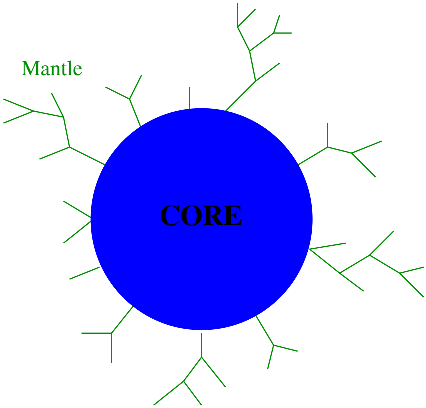
- Given graph $G = (V, E)$, imagine cutting V into two subsets V_1 and V_2 (w.l.o.g. let $|V_1| \leq |V_2|$).
- Expansion of this cut is

$$h = \frac{|(u, v) \in E : u \in V_1, v \in V_2|}{|V_1|},$$

  i.e., # of cuts per vertex.
- If in a sequence of graphs of increasing size, expansion of all cuts is bounded below by a constant, these are known as expander graphs.

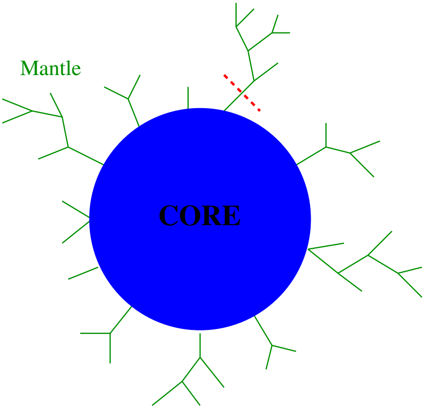
Expander Core of Giant Component

- Giant component is not an expander: cutting the largest tree gives expansion $h \sim 1/\log n$.



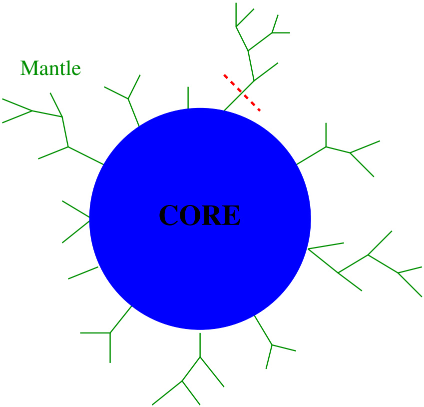
Expander Core of Giant Component

- Giant component is not an expander: cutting the largest tree gives expansion $h \sim 1/\log n$.



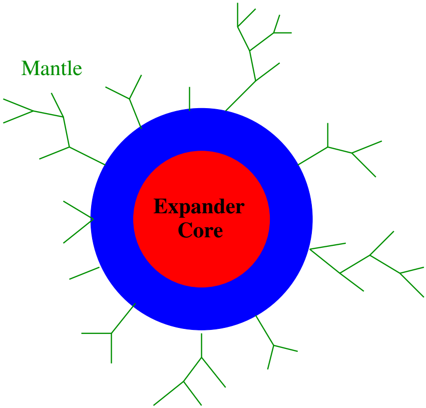
Expander Core of Giant Component

- Giant component is not an expander: cutting the largest tree gives expansion $h \sim 1/\log n$.
- But it is a "decorated expander" with an identifiable expander core. [Benjamini et al, 2006].



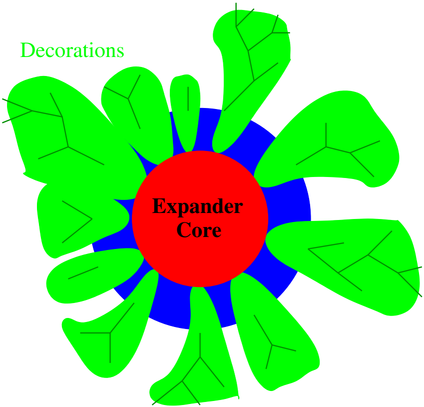
Expander Core of Giant Component

- Giant component is not an expander: cutting the largest tree gives expansion $h \sim 1/\log n$.
- But it is a "decorated expander" with an identifiable expander core. [Benjamini et al, 2006].



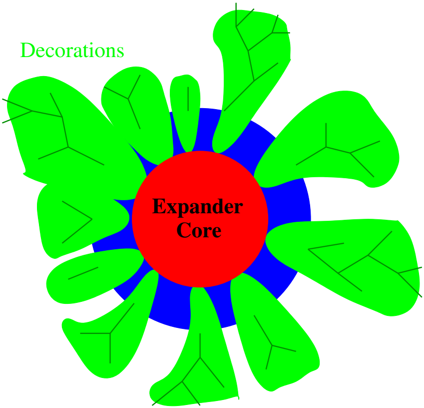
Expander Core of Giant Component

- Giant component is not an expander: cutting the largest tree gives expansion $h \sim 1/\log n$.
- But it is a "decorated expander" with an identifiable expander core. [Benjamini et al, 2006].



Expander Core of Giant Component

- Giant component is not an expander: cutting the largest tree gives expansion $h \sim 1/\log n$.
- But it is a "decorated expander" with an identifiable expander core. [Benjamini et al, 2006].
- Decorations have certain tree-like properties, and are of size $O(\log n)$.



Optimal Cut Avoids Expander Core

Claim

There exists an $\alpha_d > 2\log 2$ *such that for all* $\alpha < \alpha_d$*, an optimal bisection cannot cut any finite fraction of the expander core.*

Optimal Cut Avoids Expander Core

Claim

There exists an $\alpha_d > 2\log 2$ such that for all $\alpha < \alpha_d$, an optimal bisection cannot cut any finite fraction of the expander core.

Idea:

- Let $\epsilon = \alpha - 2\log 2$. From superlinearity of optimal bisection width, $w/\epsilon n \to 0$ as $\epsilon \to 0$.
- Number of vertices cut from giant component $\sim \epsilon n$, so optimal cut requires arbitrarily small expansion.
- Expander core cannot have cuts with vanishing expansion, so for ϵ below some constant, optimal cut must avoid expander core.

Apparent Consequences: Solution Structure

- For all $\alpha < \alpha_d$, optimal bisections only cut decorations.
- Since decorations are small, similar arguments apply as for $\alpha < 2 \log 2$: any two optimal bisections are connected by a chain of small swaps preserving balance constraint.
- All solutions then lie in a single cluster up to α_d.
- Suggests that unlike in graph coloring, $\alpha_d > \alpha_c$! This would be first known example where single cluster persists through and beyond critical threshold.

Apparent Consequences: Algorithmic Complexity

- For $\alpha < \alpha_d$, optimal bisection can be found by ranking expansion of decorations.
- As in tree-cutting upper bound, cut decorations in increasing order of expansion until giant component is pruned to size $n/2$.
- Decorations can be found in polynomial time [Benjamini et al, 2006].
- Difficulty is that unlike for trees, it could be best to cut a decoration in the middle.
- But decorations are small ($O(\log n)$), and deciding where to cut a given decoration is primarily a bookkeeping operation: takes $2^{O(\log n)} = n^{O(1)}$ operations.

Apparent Consequences: Algorithmic Complexity

Conjecture

For graphs with mean degree $\alpha < \alpha_d$ in $\mathcal{G}_{np}$, there exists an algorithm that finds the optimal bisection, w.h.p., in polynomial time.

If this conjecture is proven, it will provide a striking example of an NP-hard problem where typical instances near the phase transitions are not hard.

Final Messages

- Studying the phase structure of random combinatorial optimization problems leads to an improved understanding of typical-case algorithmic complexity.
- Analytical results on graph bisection tell us that the story is far from over: here, the hardest instances do not appear to be concentrated at the phase boundary.
- All of this analysis is for $\mathcal{G}_{np}$ graphs. Analyzing ensembles of more realistic graphs, such as those with geometric structure, remains largely an open problem.