A Distributed Algorithm for Finding Hamiltonian Cycles in Random Graphs in $O(\log n)$ Time

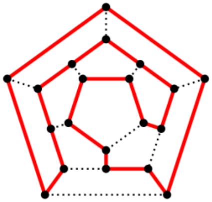
Volker Turau

25th Int. Colloquium on Structural Information and Communication Complexity
July 19th, 2018

Institute of Telematics
Hamburg University of Technology

TUHH

Hamiltonian Cycles



Definition (Hamiltonian Cycle)

A Hamiltonian cycle of an undirected graph G is a cycle through G that visits each node exactly once.

- Corresponding decision problem is NP-complete
- We consider random graphs

Theorem (Komlós & Szemerédi, 1983)

$G(n,p)$ contains w.h.p. a Hamiltonian cycle, provided

$$p \geq p_{crit} = (\log n + \log\log n + \omega(n))\,/\,n$$

where $\lim_{n\to\infty} \omega(n) = \infty$.

Finding Hamiltonian cycles in $G(n, p)$

- Deterministic sequential algorithms
  - Bollobás, Fenner & Frieze, 1987:
    $O(n^{3+o(1)})$ algorithm that works w.h.p. at threshold p_{crit}
  - Frieze & Haber, 2015:
    $O(n^{1+o(1)})$ algorithm that works w.h.p. if $\delta(G) \geq 3$
- It is a non-local graph problem, i.e., it is required to always consider the entire graph in order to solve the problem

Finding Hamiltonian cycles in $G(n, p)$

- Synchronous distributed algorithms
  - Chatterjee et al., 2018:
    If $p \geq c \log n / \sqrt{n}$ then w.h.p. a Hamiltonian cycle can be found in $\tilde{O}(\sqrt{n})$ rounds
  - Ghaffari and Li, 2018:
    If $p \geq C \log n / n$ and nodes have unlimited memory then w.h.p. a Hamiltonian cycle can be found in $2^{O(\sqrt{\log n})}$ rounds
- Our result

Theorem

Let $G(n, p)$ with $p \geq (\log n)^{3/2} / \sqrt{n}$ be a random graph. Algorithm $\mathcal{A}_{\mathrm{HC}}$ computes in the synchronous model w.h.p. a Hamiltonian cycle for G terminating in $O(\log n)$ rounds . It uses messages of size $O(\log n)$ and $O(\log n)$ memory per node.

Computational Model & Assumptions

- Synchronous $\mathcal{CONGEST}$ model, i.e. messages of size $O(\log n)$
- Each node has $O(\log n)$ local memory
- A distinguished node v_0
- Results of this work hold *with high probability* (w.h.p.) which means with probability tending to 1 as $n \to \infty$

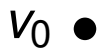
1

Informal Description of $\mathcal{A}_{\mathrm{HC}}$

Algorithm $\mathcal{A}_{\mathrm{HC}}$

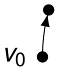
- $\mathcal{A}_{\mathrm{HC}}$ works in phases
  - First phase ($3 \log n$ rounds)
    - Starting in v_0 a path P of length $3 \log n$ is built
  - Second phase ($3 \log n$ rounds)
    - Path P is closed to a cycle C of length at most $4 \log n$
  - Middle phases
    - Concurrently edges are replaced by two edges
    - After $16 \log n$ phases C has w.h.p. at least $n - 3 \log n$ nodes
  - Final phases
    - Each final phase integrates one node into C
- Each middle and final phase lasts a constant number of rounds

Phases



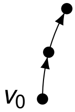
Phase 0

Phases



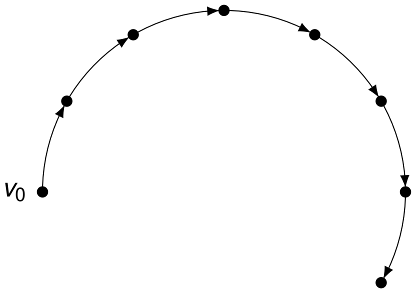
Phase 0

Phases



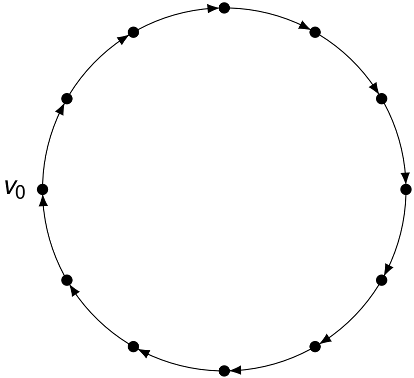
Phase 0

Phases



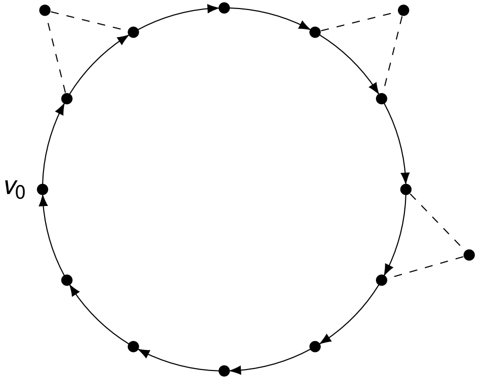
After $3 \log n$ rounds

Phases



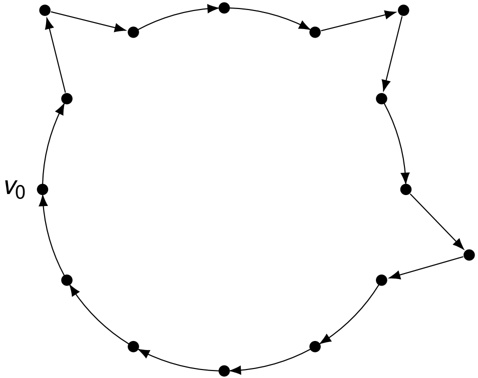
Phase 1, after another $3 \log n$ rounds

Phases



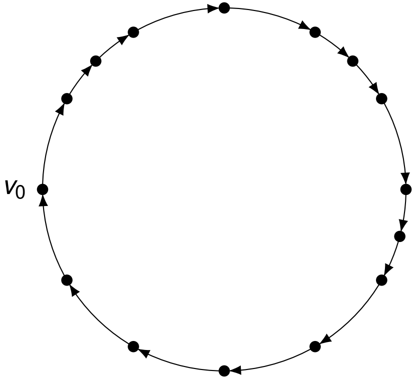
Middle Phase Step 1

Phases



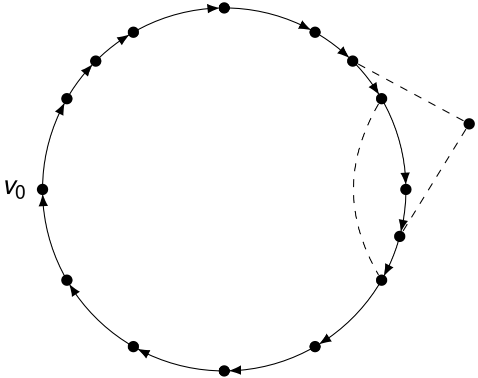
Middle Phase Step 2

Phases



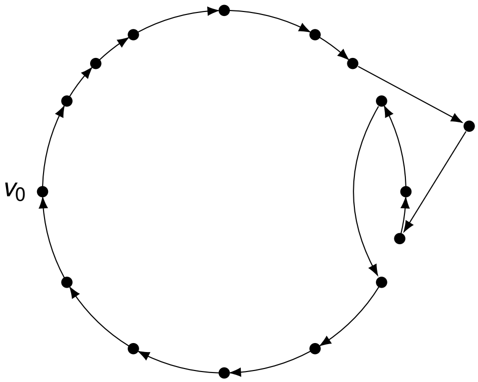
After Middle Phases

Phases



Final Phase Step 1

Phases

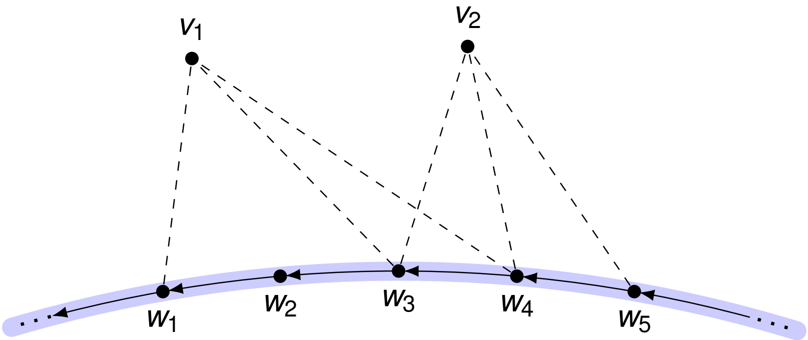


Final Phase Step 2

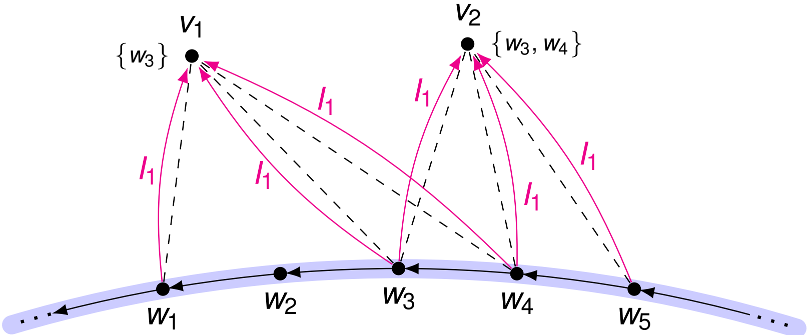
2

Details of $\mathcal{A}_{\mathrm{HC}}$

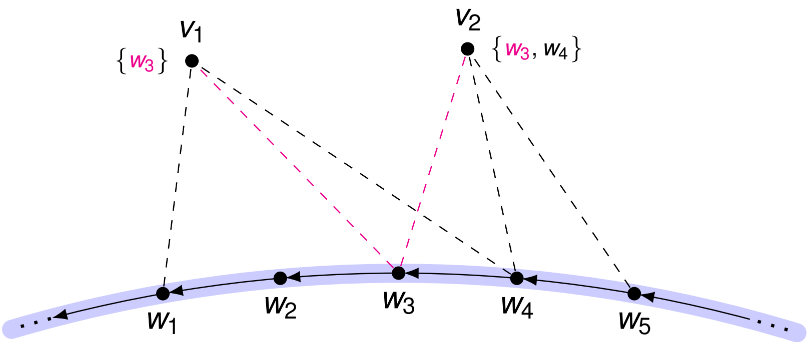
Middle Phases



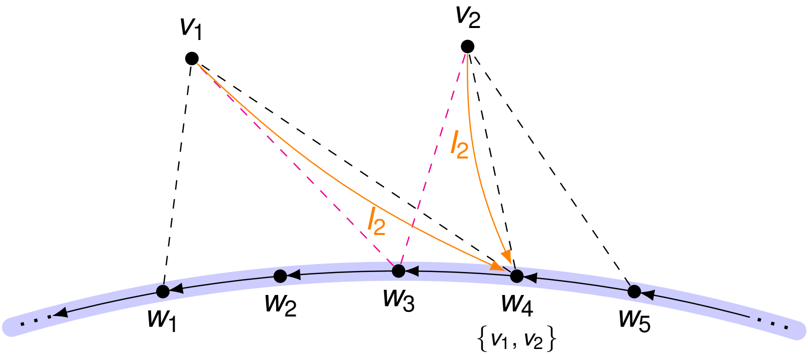
Middle Phases



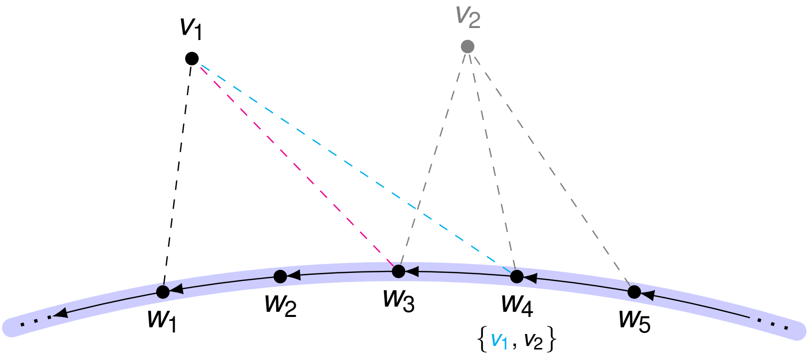
Middle Phases



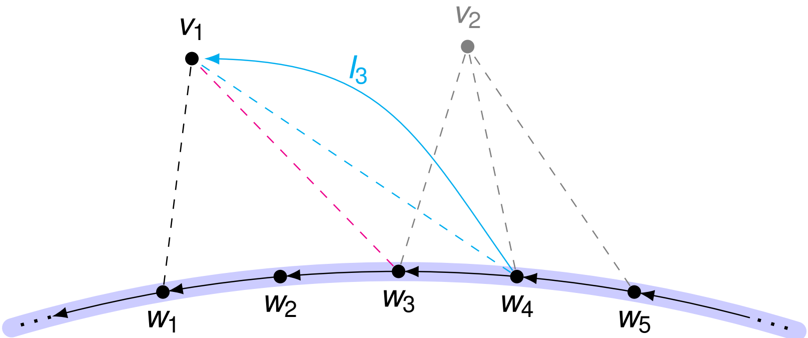
Middle Phases



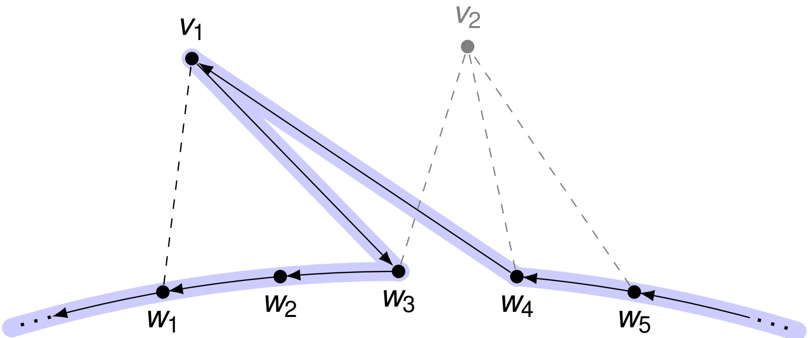
Middle Phases



Middle Phases



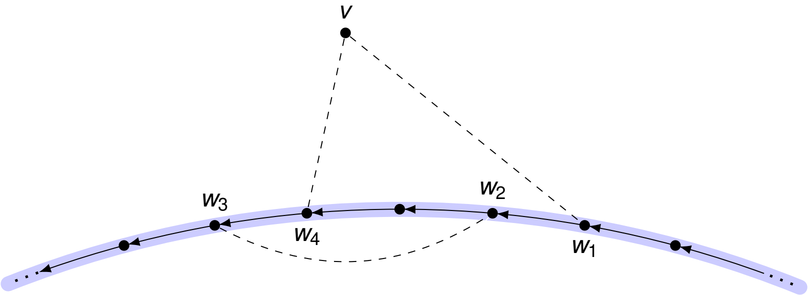
Middle Phases



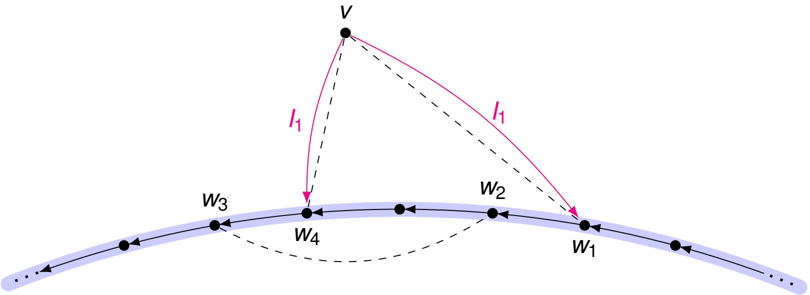
Observations

- Individual extensions do not interfere with each other because
  - Each node outside C sends in each middle phase at most one request to integrate and
  - each edge of C accepts at most one integration request
- After at most $16 \log n$ middle phases C has w.h.p. at least $n - 3 \log n$ nodes

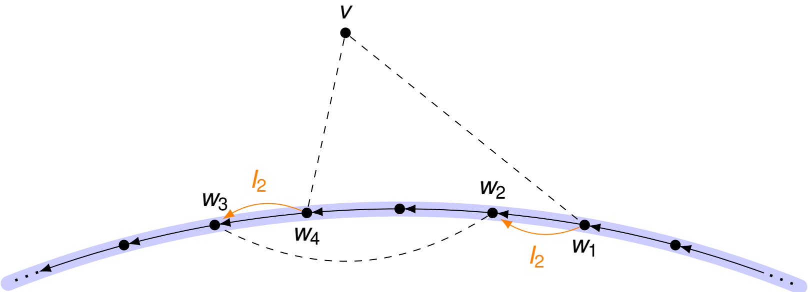
Final Phases



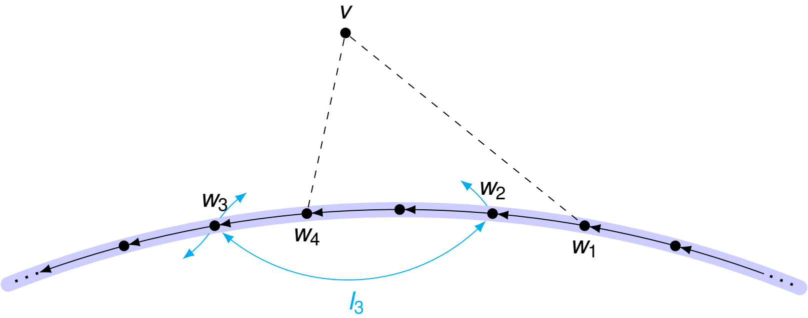
Final Phases



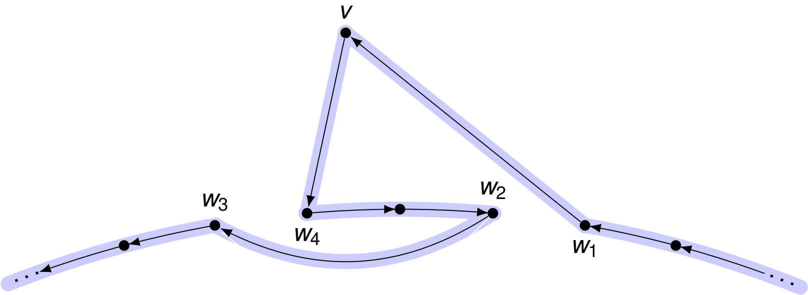
Final Phases



Final Phases

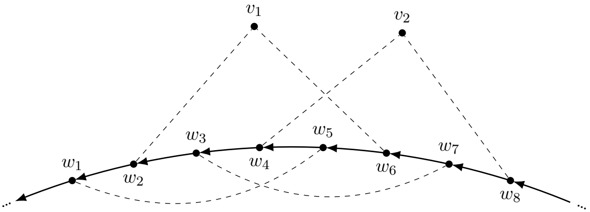


Final Phases

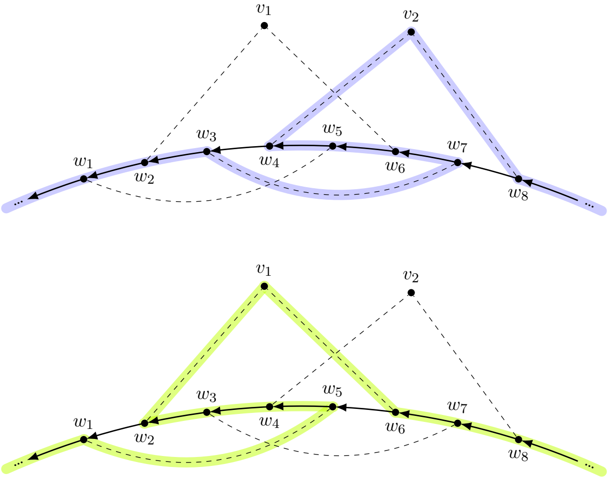


Final Phases

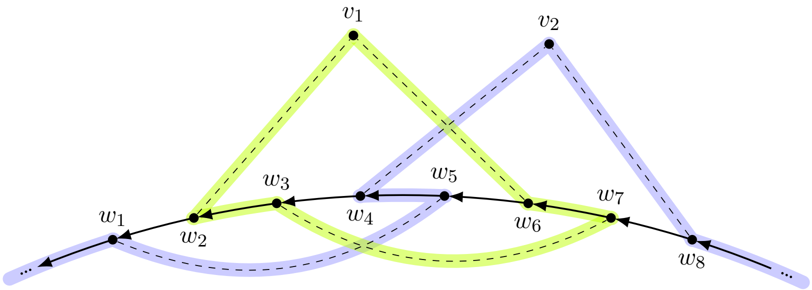
- Observation 1: Integration steps cannot be executed concurrently
  - If segments, which are inverted overlap, separate cycles may occur



Final Phases



Final Phases

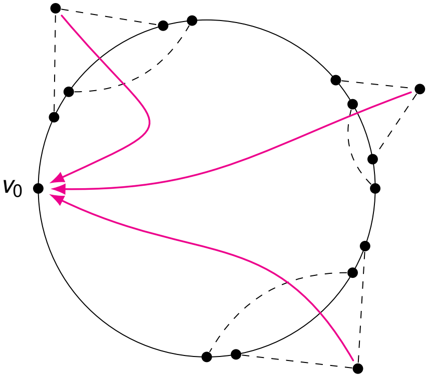


Final Phases

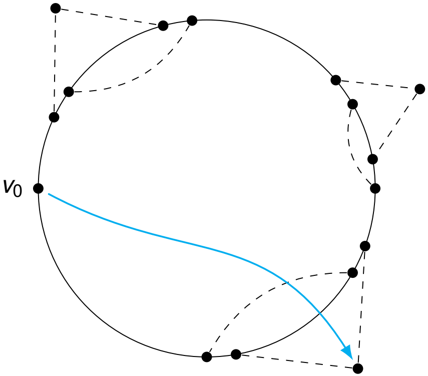
Solution:

- Final phases are done sequentially
- Node v_0 acts as a coordinator
- All nodes from $V \setminus C$ that can be integrated report this to v_0
- v_0 randomly selects one node to be integrated and informs it
- This requires a short route from each node to v_0
  - A pre-processing phase builds a BFS-tree rooted in v_0
  - Note that diameter is at most 3 because $p \geq \sqrt{\log n / n}$

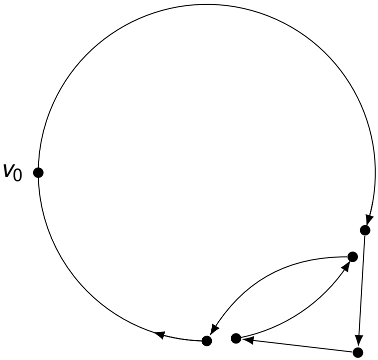
Final Phases



Final Phases



Final Phases

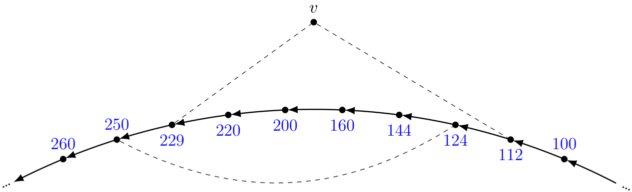


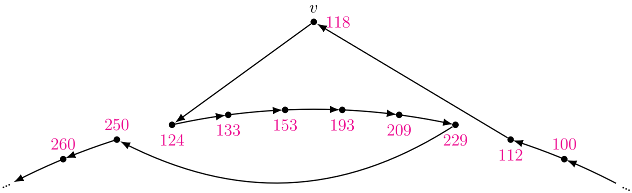
Final Phases

- Observation 2: Direction of Edges between w_2 and w_4 must be inverted in a constant number of rounds
  - Sequentially reversing is no option (unbounded number of edges)

- Solution
  - Nodes on C carry a number, strictly increasing beginning with v_0
  - Numbering in phases 0 and 1: $n^{14}, 2n^{14}, 3n^{14}, \ldots$
  - Middle phases
    - A node integrated between nodes with numbers $l < r$ gets number $\lceil (l + r)/2 \rceil$
  - Final phases
    - After v_0 decides the node to integrate, it broadcasts numbers l and r (of w_2 and w_4) into graph
    - Nodes with number $l \leq x \leq r$ get the new number $l + r - x$ and reverse corresponding edge

Final Phases





3

Complexity of $\mathcal{A}_{\mathrm{HC}}$

Preserving the Required Randomness

- Iterative algorithms on random graphs must be organized such that one only slowly uncovers the random choices in input graph
- For $\hat{p} = 1 - (1-p)^{1/\gamma \log n}$ graph $G(n,p)$ is equal to union of $\gamma \log n$ independent copies of $G(n,\hat{p})$
- Since $\hat{p} \geq \sqrt{\log n}/\gamma\sqrt{n}$ we have

$$\bigcup_{i=1}^{\gamma \log n} G(n,q) \subseteq G(n,p)$$

with $q = \sqrt{\log n}/\gamma\sqrt{n}$

Preserving the Required Randomness

- For $i \geq 0$ let G^i be the union of i independent copies of $G(n, q)$
- Cycle C of phase i consists of edges belonging to G^i
- Probability that any two nodes of V are connected with an edge from $G^{i+1} \setminus G^i$ is q

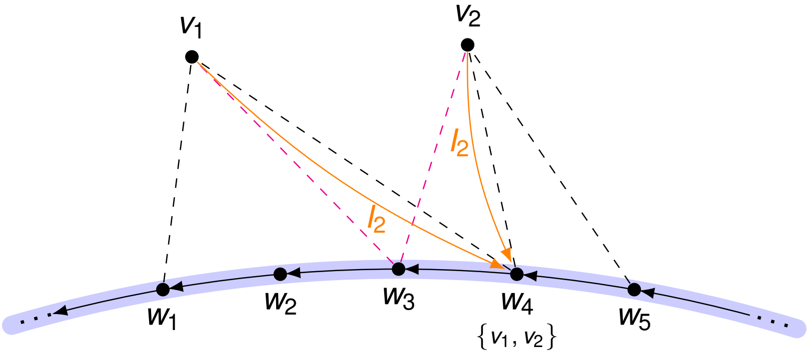
- Thus, in each phase a new copy of $G(n, q)$ is revealed

Complexity of Middle Phases

- How many nodes are integrated into C per middle phase?
  - How many nodes on C send an accept message I_3?
  - How many nodes outside C send an invitation I_2?

Complexity of Middle Phases

- How many nodes outside C send an invitation I_2?



Complexity of Middle Phases

- A node $v \in V \setminus C$ sends an invitation if it is connected to at least one pair of consecutive nodes on C
- This event has probability q^2, but these events are not independent

- Event π_v: $v \in V \setminus C$ forms a triangle with at least one of every second edge of C
- π_v has probability $1 - (1 - q^2)^{c/2}$ and π_v's are independent (c number of nodes on C)

Random Variable X

X is the number of nodes v where π_v occurs

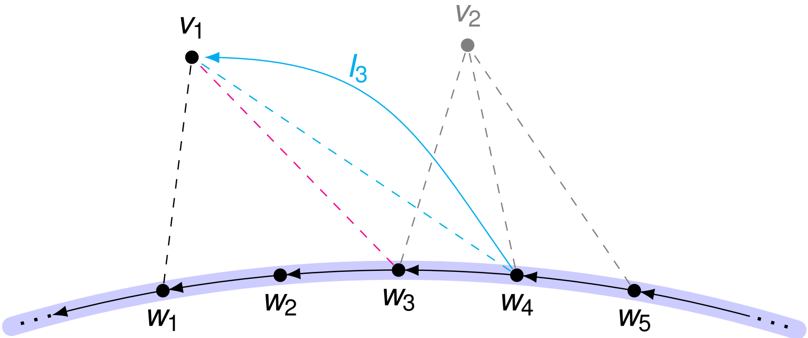
Complexity of Middle Phases

- X is a lower bound for number of nodes that send invitation I_2.

$$E[X] = (n - c)(1 - (1 - q^2)^{c/2}) \tag{1}$$

Complexity of Middle Phases

- How many nodes on C send an accept message l_3?



Complexity of Middle Phases

Random variable Y

Number of nodes that are integrated into C in a middle phase.

- Computation of Y can be reduced to bins and balls model
  - X number of balls; c number of bins
  - Each ball is thrown randomly in any of the c bins
  - Probability that $v \in C$ is connected to w in $V \setminus C$ is independently of v and w equal to p.
- Y is equal to number of nonempty bins

$$E[Y|X = x] = c\left(1 - \left(1 - \frac{1}{c}\right)^x\right) \tag{2}$$

Complexity of Middle Phases

- We need lower bounds for X and Y
- Use Chernoff bound
- We distinguish the cases $c < n/7$ and $c \geq n/7$
- Reason: Variance of X behaves differently in these two ranges
  - $c < n/7$: Variance is rather large
  - $c \geq n/7$: Variance tends to 0

The case $c < n/7$

Lemma

If $3 \log n < c < n/7$ *then* $X > c/3$ *w.p.* $1 - 1/n^d$ *for some* $d > 0$.

Lemma

If $3 \log n < c < n/7$ *then w.h.p.* $\frac{Y}{c} \geq 0.92 \left(1 - \frac{1}{e^{1/3}}\right)$.

Lemma

Let C *be a cycle with at least* $3 \log n$ *nodes. Then after at most* $3 \log n$ *phases* C *has w.h.p. at least* $n/7$ *nodes.*

The case $c \geq n/7$

Lemma

Let C be a cycle with at least $n/7$ nodes. Then after $13 \log n$ middle phases C has w.h.p. at least $n - 3 \log n$ nodes.

- The last two lemma show that w.h.p. after $16 \log n$ phases C has w.h.p. at least $n - 3 \log n$ nodes

Complexity of Final Phases

Lemma

Let $q \geq \sqrt{\log n / n}$. In each final phase w.h.p. a node $v \in V \setminus C$ is integrated into C.

4

Conclusion

Conclusion & Outlook

- Algorithm $\mathcal{A}_{\mathrm{HC}}$ computes in $O(\log n)$ rounds w.h.p. a Hamiltonian cycle for a random graph $G(n,p)$ provided $p \geq (\log n)^{3/2}/\sqrt{n}$

- By maxing out arguments of paper it may be possible to prove result for $p = \sqrt{\log n/n}$
- What about p closer to $p_{crit} = (\log n + \log\log n + \omega(n))/n$?
- Some of our arguments cannot be applied if $p = 1/\sqrt{n}$ let alone $p = p_{crit}$
- We suspect that finding a distributed $O(\log n)$ round algorithm for $p \in o(1/\sqrt{n})$ is a hard task

A Distributed Algorithm for Finding Hamiltonian Cycles in Random Graphs in $O(\log n)$ Time

25th Int. Colloquium on Struc ... plexity

Volker Turau

Professor

Phone +49 / (0)40 428 78 3530

e-Mail turau@tuhh.de

http://www.ti5.tu-harburg.de/staff/turau

Institute of Telematics
Hamburg University of Technology

TUHH