

Power Control in Ad Hoc Wireless Network Fault-Tolerant Design

Celso C. Ribeiro
Renato N. Moraes

Universidade Federal Fluminense, Brazil

June 2009

- 1 Wireless ad hoc networks
 - Definition
 - Constraints
- 2 Problem statement
- 3 System model and related work
 - Variants
- 4 Integer programming formulation
 - Variables
 - Model
- 5 GRASP heuristic
 - Construction phase
 - Local search phase
- 6 Numerical results
 - Experimental settings
 - Optimal solutions
 - GRASP approximate solutions
- 7 Concluding remarks

- 1 Wireless ad hoc networks
 - Definition
 - Constraints
- 2 Problem statement
- 3 System model and related work
 - Variants
- 4 Integer programming formulation
 - Variables
 - Model
- 5 GRASP heuristic
 - Construction phase
 - Local search phase
- 6 Numerical results
 - Experimental settings
 - Optimal solutions
 - GRASP approximate solutions
- 7 Concluding remarks

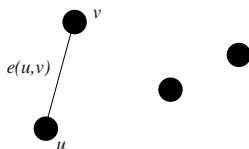
Definition

An *ad hoc network* consists of a collection of transceivers, in which a packet may have to traverse multiple consecutive wireless links to reach its final destination.

- ▶ Ad hoc networks can be represented by a set V of transceivers (nodes) together with their locations or the distances between them.

Wireless ad hoc networks

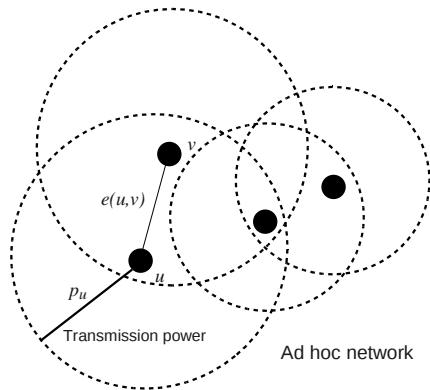
- ▶ For each ordered pair (u, v) of transceivers, with $u, v \in V$, we are given a non-negative arc weight $e(u, v) = d_{uv}^\varepsilon$
 - ▶ d_{uv} is the Euclidean distance between the transmitter u and the receiver v
 - ▶ ε is the loss exponent, typically equal to two.



Ad hoc network

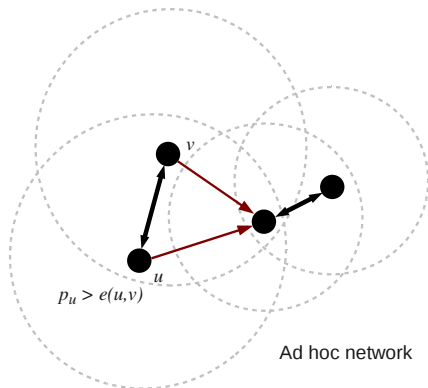
Wireless ad hoc networks

- ▶ A transmission power p_u is associated with each node $u \in V$.



Wireless ad hoc networks

- ▶ A signal transmitted by the transceiver u can be received at node v if and only if the transmission power of u is at least equal to $e(u, v)$, i.e. if $p_u \geq e(u, v)$.



- ▶ Wireless networks face a variety of constraints that do not appear in wired networks.
- ▶ Power constraints:
 - ▶ Nodes are battery powered: it is expensive and sometimes even infeasible to recharge the device.
 - ▶ Radios tend to be the major source of power dissipation.
 - ▶ Instead of transmitting with maximum power, algorithms adjust the transmission power of each node.
- ▶ Connectivity constraints:
 - ▶ Fault-tolerance requirements, due to their critical application domains and to the large number of failures.
 - ▶ If there is only one path between a pair of nodes, failure of a node or link between them will result in a disconnected graph.
 - ▶ Topologies with multiple, disjoint paths between any pair of nodes are often required.

- ▶ Wireless networks face a variety of constraints that do not appear in wired networks.
- ▶ Power constraints:
 - ▶ Nodes are battery powered: it is expensive and sometimes even infeasible to recharge the device.
 - ▶ Radios tend to be the major source of power dissipation.
 - ▶ Instead of transmitting with maximum power, algorithms adjust the transmission power of each node.
- ▶ Connectivity constraints:
 - ▶ Fault-tolerance requirements, due to their critical application domains and to the large number of failures.
 - ▶ If there is only one path between a pair of nodes, failure of a node or link between them will result in a disconnected graph.
 - ▶ Topologies with multiple, disjoint paths between any pair of nodes are often required.

- ▶ Wireless networks face a variety of constraints that do not appear in wired networks.
- ▶ Power constraints:
 - ▶ Nodes are battery powered: it is expensive and sometimes even infeasible to recharge the device.
 - ▶ Radios tend to be the major source of power dissipation.
 - ▶ Instead of transmitting with maximum power, algorithms adjust the transmission power of each node.
- ▶ Connectivity constraints:
 - ▶ Fault-tolerance requirements, due to their critical application domains and to the large number of failures.
 - ▶ If there is only one path between a pair of nodes, failure of a node or link between them will result in a disconnected graph.
 - ▶ Topologies with multiple, disjoint paths between any pair of nodes are often required.

- 1 Wireless ad hoc networks
 - Definition
 - Constraints
- 2 Problem statement
- 3 System model and related work
 - Variants
- 4 Integer programming formulation
 - Variables
 - Model
- 5 GRASP heuristic
 - Construction phase
 - Local search phase
- 6 Numerical results
 - Experimental settings
 - Optimal solutions
 - GRASP approximate solutions
- 7 Concluding remarks

- ▶ Given the node set V and non-negative arc weights $e(u, v)$ for any $u, v \in V$:

Definition

The *biconnected minimum power consumption problem* consists of finding an optimal assignment of transmission powers $p : V \rightarrow R_+$ to every node $u \in V$, such that the total power consumption $\sum_{u \in V} p_u$ is minimized and the resulting transmission graph $G = (V, E)$, where $E = \{(u, v) : u \in V, v \in V, p_u \geq e(u, v)\}$, is biconnected.

- ▶ This problem was proved to be NP-hard.

- 1 Wireless ad hoc networks
 - Definition
 - Constraints
- 2 Problem statement
- 3 System model and related work
 - Variants
- 4 Integer programming formulation
 - Variables
 - Model
- 5 GRASP heuristic
 - Construction phase
 - Local search phase
- 6 Numerical results
 - Experimental settings
 - Optimal solutions
 - GRASP approximate solutions
- 7 Concluding remarks

- ▶ Set V of transceivers, with $|V| = n$, each of them equipped with an omnidirectional antenna which is responsible for sending and receiving signals.
- ▶ Ad hoc network established by assigning a transmission power p_u to each transceiver $u \in V$.
- ▶ Each node can (possibly dynamically) adjust its transmitting power, based on the distance to the receiving nodes and on the background noise.

- ▶ Power requirement at node u for supporting transmissions through a link from u to v :

$$e(u, v) \geq d_{uv}^\varepsilon \cdot q_v,$$

- ▶ where:
 - ▶ d_{uv} is the Euclidean distance between the transmitter u and the receiver v ,
 - ▶ ε is the loss exponent, and
 - ▶ q_v is the receiver's power threshold for signal detection, usually normalized to one.

- ▶ Power requirement for supporting transmissions between nodes u and v are symmetric:

$$e(u, v) = e(v, u)$$

- ▶ Symmetric version widely accepted as reasonable:
 - ▶ Holds for free-space environments with non-obstructed lines of sight.
 - ▶ Disregards reflections, scattering, and diffraction caused e.g. by buildings and terrains.

- ▶ There may exist pairs of transceivers $u, v \in V$ such that

$$e(u, v) \neq e(v, u)$$

- ▶ Some situations:
 - ▶ Batteries with different power levels.
 - ▶ Heterogeneous nodes.
 - ▶ Different levels of ambient noise in the regions containing the two nodes.

- ▶ Communication from u to v enabled whenever $p_u \geq e(u, v)$.
- ▶ Transmission graph $G = (V, E)$ associated with a power assignment $p : V \rightarrow R_+$:

$$E = \{(u, v) : u \in V, v \in V, p_u \geq e(u, v)\}$$

- ▶ All arcs established by the power settings are considered to enforce the biconnectedness.

System model: bidirectional topology

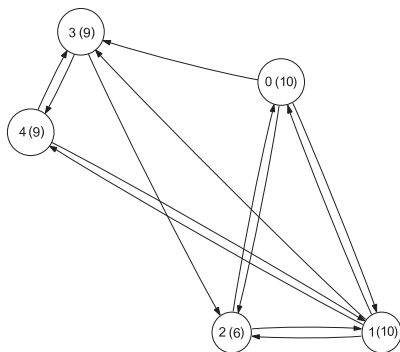
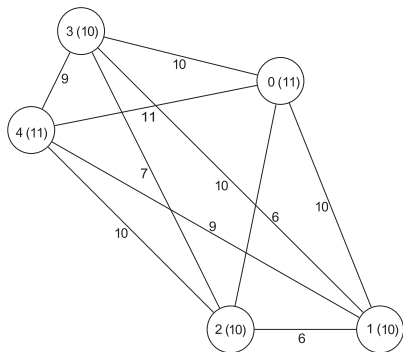
- ▶ Communication from u to v enabled whenever $p_u \geq e(u, v)$ and $p_v \geq e(v, u)$.
- ▶ Restricted arc set considered to enforce the biconnectivity constraints.
- ▶ Transmission graph $G = (V, B)$ associated with a power assignment $p : V \rightarrow R_+$:

$$B = \{(u, v) : u \in V, v \in V, p_u \geq e(u, v), p_v \geq e(v, u)\} \subseteq E$$

- ▶ Edge $[u, v]$ is used as a communication link to enforce biconnectedness if v is within the transmission range of u and u is within the transmission range of v .

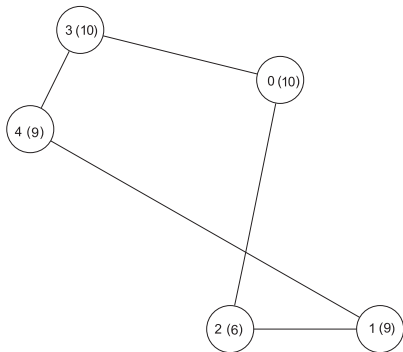
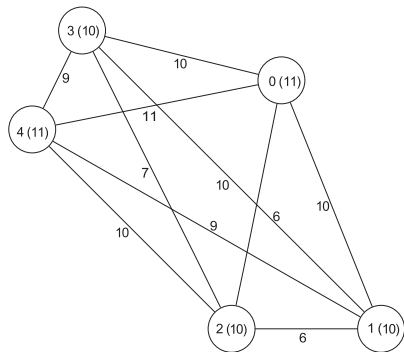
System model: example with symmetric input

- ▶ Minimum biconnected unidirectional topology solution
- ▶ $G(p) = (V, E(p))$
- ▶ Total power consumption: 44



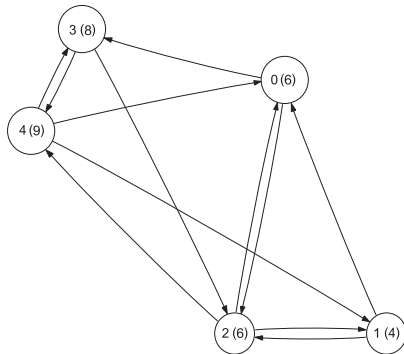
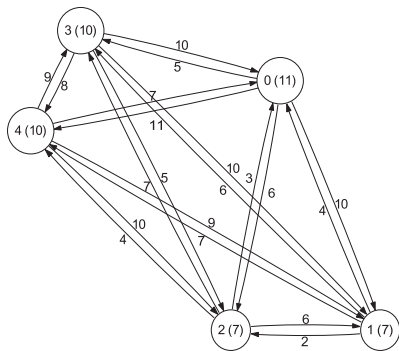
System model: example with symmetric input

- ▶ Minimum biconnected bidirectional topology solution
- ▶ $G(p) = (V, B(p))$
- ▶ Total power consumption: 45



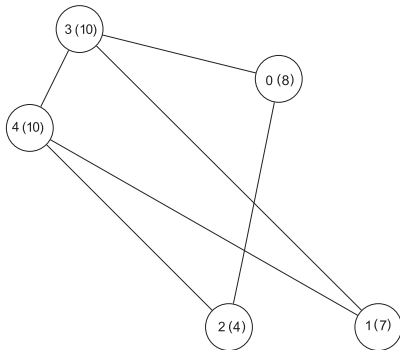
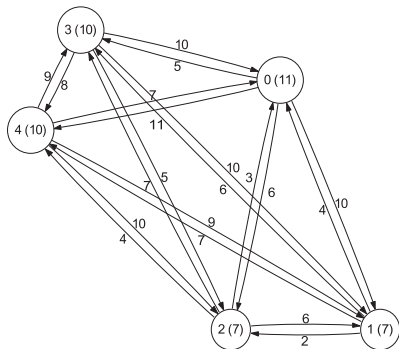
System model: example with asymmetric input

- ▶ Minimum biconnected unidirectional topology solution
- ▶ $G(p) = (V, E(p))$
- ▶ Total power consumption: 33



System model: example with asymmetric input

- ▶ Minimum biconnected bidirectional topology solution
- ▶ $G(p) = (V, B(p))$
- ▶ Total power consumption: 39



- ▶ Four versions of the biconnected minimum power consumption problem:
 - ▶ Symmetric input with unidirectional topology
 - ▶ Symmetric input with bidirectional topology
 - ▶ Asymmetric input with unidirectional topology
 - ▶ Asymmetric input with bidirectional topology

Unidirectional topology

- ▶ Connected transmission graph
 - ▶ [Chen and Huang, 1989](#):
 - ▶ NP-hardness
 - ▶ 2-approximation algorithm
 - ▶ [Kirousis et al., 2000](#):
 - ▶ NP-hardness in three-dimensional Euclidean space
 - ▶ 2-approximation algorithm
- ▶ Biconnected transmission graph
 - ▶ [Calinescu and Wan, 2006](#):
 - ▶ NP-hardness
 - ▶ 4-approximation algorithm

Bidirectional topology

- ▶ Connected transmission graph
 - ▶ Calinescu et al., 2002:
 - ▶ NP-completeness
 - ▶ Althaus et al., 2006:
 - ▶ $(5/3 + \epsilon)$ -approximation algorithm
 - ▶ Branch-and-cut algorithm
- ▶ Biconnected transmission graph
 - ▶ Lloyd et al., 2005:
 - ▶ $(2(2 - 2/n)(2 + 1/n))$ -approximation algorithm

Unidirectional topology

- ▶ Connected transmission graph
 - ▶ [Krumke et al., 2003](#), [Calinescu et al., 2003](#), [Caragiannis et al., 2006](#):
 - ▶ $O(\log n)$ -approximation algorithm

Bidirectional topology

- ▶ Connected transmission graph
 - ▶ [Caragiannis et al., 2006](#):
 - ▶ $O(1.35 \log n)$ -approximation algorithm
 - ▶ [Calinescu et al., 2003](#):
 - ▶ $O(\log n)$ -approximation algorithm

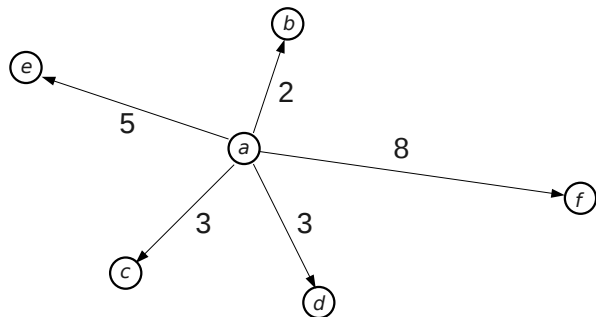
- 1 Wireless ad hoc networks
 - Definition
 - Constraints
- 2 Problem statement
- 3 System model and related work
 - Variants
- 4 Integer programming formulation**
 - Variables**
 - Model**
- 5 GRASP heuristic
 - Construction phase
 - Local search phase
- 6 Numerical results
 - Experimental settings
 - Optimal solutions
 - GRASP approximate solutions
- 7 Concluding remarks

Integer programming formulation: variables

- ▶ Multicommodity network flow model (Magnanti and Raghavan, 2005)
- ▶ C is a set of $\lceil |V|/2 \rceil$ commodities.
- ▶ For each commodity $c \in C$:
 - ▶ $o(c)$ is its origin
 - ▶ $d(c)$ is its destination
- ▶ For any node $i \in V$ and any commodity $c \in C$:
 - ▶ $D_c(i) = -2$ if $i = o(c)$,
 - ▶ $D_c(i) = +2$ if $i = d(c)$,
 - ▶ $D_c(i) = 0$ otherwise.
- ▶ For node $i \in V$ and commodity $c \in C$, the binary variable f_{ij}^c represents the flow of commodity c through arc (i, j) :
 - ▶ $f_{ij}^c = 1$, if arc (i, j) is used by commodity c for communication from node i to j ,
 - ▶ $f_{ij}^c = 0$, otherwise.

- ▶ $P_i = [p_i^1, \dots, p_i^{\phi(i)}]$ is a list of increasing power levels that can be assigned to node $i \in V$, where:
 - ▶ p_i^1 is the minimum power such that transmissions from i reach at least one node in $V \setminus \{i\}$
 - ▶ $p_i^{\ell+1} > p_i^\ell$ for $\ell = 1, \dots, \phi(i) - 1$
 - ▶ $p_i^\ell > 0$ for $\ell = 1, \dots, \phi(i) - 1$
 - ▶ $p_i^0 = 0$ for ease of representation
- ▶ $T_i^\ell \neq \emptyset$ is the set of new nodes reachable from i if its power level increases from $p_i^{\ell-1}$ to p_i^ℓ , for $\ell = 1, \dots, \phi(i)$.

Integer programming formulation: variables



- ▶ $P_a = [2, 3, 5, 8]$
- ▶ $T_a^1 = \{b\}$, $T_a^2 = \{c, d\}$, $T_a^3 = \{e\}$, $T_a^4 = \{f\}$

- ▶ Nodes $i \in V$ and power levels $P_i = [p_i^1, \dots, p_i^{\phi(i)}]$
- ▶ Binary variables x_i^ℓ determine the power level assigned to each node $i \in V$:
 - ▶ $x_i^\ell = 1$, if there is a node $j \in T_i^\ell$ such that link (i, j) is used for communication from i to j ,
 - ▶ $x_i^\ell = 0$, otherwise.
- ▶ $p_i^{\bar{\ell}(i)}$ is the minimum power level such that transmissions from i reach at least two nodes in $V \setminus \{i\}$.

Integer programming formulation: model

$$\min \sum_{i \in V} \sum_{\ell=1}^{\phi(i)} (p_i^\ell - p_i^{\ell-1}) \cdot x_i^\ell$$

subject to:

$$\sum_{j \in V} f_{ji}^c - \sum_{l \in V} f_{il}^c = D_c(i), \quad \forall c \in C, \forall i \in V$$

$$\sum_{j \in V} f_{ij}^c \leq 1, \quad \forall c \in C, \forall i \in V : i \neq o(c), i \neq d(c)$$

$$x_i^\ell \geq f_{ij}^c, \quad \forall i \in V, \forall c \in C, \forall j \in T_i^\ell, \ell = 1, \dots, \phi(i)$$

$$x_i^{\ell+1} \leq x_i^\ell, \quad \forall i \in V, \ell = 1, \dots, \phi(i) - 1$$

$$x_i^\ell = 1, \quad \forall i \in V, \ell = 1, \dots, \bar{\ell}(i)$$

$$f_{ij}^c \in \{0, 1\}, \quad \forall i, j \in V, \forall c \in C$$

$$x_i^\ell \in \{0, 1\}, \quad \forall i \in V, \ell = 1, \dots, \phi(i)$$

- ▶ Flow conservation equations:

$$\sum_{j \in V} f_{ji}^c - \sum_{l \in V} f_{il}^c = D_c(i), \quad \forall c \in C, \forall i \in V$$

- ▶ Node-disjointness:

$$\sum_{j \in V} f_{ij}^c \leq 1, \quad \forall c \in C, \forall i \in V : i \neq o(c), i \neq d(c)$$

- ▶ $x_i^\ell = 1$ if there is a node $j \in T_i^\ell$ such that arc (i, j) is used for communication from i to j by commodity c :

$$x_i^\ell \geq f_{ij}^c, \quad \forall i \in V, \forall c \in C, \forall j \in T_i^\ell, \ell = 1, \dots, \phi(i)$$

- ▶ $x_i^{\ell+1} = 0$ if the previous level was not used:

$$x_i^{\ell+1} \leq x_i^{\ell}, \quad \forall i \in V, \ell = 1, \dots, \phi(i) - 1$$

- ▶ Minimum power level $\bar{\ell}(i)$ reaches at least the two closest nodes:

$$x_i^{\ell} = 1, \quad \forall i \in V, \ell = 1, \dots, \bar{\ell}(i)$$

- ▶ Integrality requirements:

$$f_{ij}^c \in \{0, 1\}, \quad \forall i, j \in V, \forall c \in C$$

$$x_i^{\ell} \in \{0, 1\}, \quad \forall i \in V, \ell = 1, \dots, \phi(i)$$

- 1 Wireless ad hoc networks
 - Definition
 - Constraints
- 2 Problem statement
- 3 System model and related work
 - Variants
- 4 Integer programming formulation
 - Variables
 - Model
- 5 GRASP heuristic**
 - Construction phase
 - Local search phase
- 6 Numerical results
 - Experimental settings
 - Optimal solutions
 - GRASP approximate solutions
- 7 Concluding remarks

- ▶ A greedy randomized adaptive search procedure (GRASP) is a multistart process.
- ▶ Each of its iterations consists of two phases:
 1. Construction: a feasible solution is built
 2. Local search (or improvement): a local optimum in the neighborhood of the current solution is sought.
- ▶ Best overall solution is returned.
- ▶ GRASP heuristic for the asymmetric input with bidirectional topology variant of the biconnected minimum power consumption problem.

Algorithm 1 Pseudo-code of a GRASP heuristic for minimization problems

Require: $Max_Iterations, Seed$

Ensure: Best known solution x^*

- 1: $f^* \leftarrow \infty$;
 - 2: **for** $iteration = 1, \dots, Max_Iterations$ **do**
 - 3: $x \leftarrow Greedy_Randomized_Construction(Seed)$;
 - 4: $x \leftarrow Local_Search(x)$;
 - 5: **if** $cost(x) < f^*$ **then**
 - 6: $x^* \leftarrow x$;
 - 7: $f^* \leftarrow cost(x)$;
 - 8: **return** x^* ;
-

- ▶ Randomized construction phase has two stages:
 1. Builds a bidirectional connected graph $H = (V, E)$
 2. Extends E to produce a biconnected graph $G = (V, B)$
- ▶ Greedy function that guides construction is based on the wireless multicast advantage property: if p_u is the current power assigned to node u and there is a node v such that $e(u, v) > p_u$, then the additional power required to set up communication from u to v is $e(u, v) - p_u$.

Greedy function for any $u, v \in V$

$$g(u, v) = \max\{0, e(u, v) - p_u\} + \max\{0, e(v, u) - p_v\}$$

- ▶ If $g(u, v) = 0$, bidirectional communication between u and v is already set up.

Algorithm 2 First stage of the randomized construction phase

Require: Node set V , initial node $r \in V$, parameter $\alpha \in [0, 1]$

Ensure: Bidirectional connected graph $H = (V, E)$

- 1: $p_u \leftarrow 0$ for all $u \in V$;
 - 2: $V' \leftarrow \{r\}$;
 - 3: $E \leftarrow \emptyset$;
 - 4: **while** $V \setminus V' \neq \emptyset$ **do**
 - 5: **for all** $u \in V \setminus V'$ **do**
 - 6: $g(u) \leftarrow \min_{v \in V'} \{g(u, v)\}$;
 - 7: $\underline{g} \leftarrow \min_{u \in V \setminus V'} \{g(u)\}$;
 - 8: $\overline{g} \leftarrow \max_{u \in V \setminus V'} \{g(u)\}$;
 - 9: Randomly select node $u \in V \setminus V'$ such that $g(u) \leq \underline{g} + \alpha(\overline{g} - \underline{g})$;
 - 10: $V' \leftarrow V' \cup \{u\}$;
 - 11: $E \leftarrow E \cup \{[u, v]\}$, with $v \in V' : g(u) = g(u, v)$;
 - 12: Update the power assignments p_u and p_v ;
 - 13: **return** $H = (V, E)$;
-

- ▶ Biconnected component of a graph: maximal subset of nodes such that there are two disjoint paths between any two of them.
- ▶ Articulation point: node belonging to more than one of its biconnected components.
- ▶ Tarjan's algorithm is used to compute the biconnected components and articulation points of the current solution.
- ▶ Pairs of biconnected components are linked one by one.
- ▶ Second stage stops when a biconnected graph is built.

Algorithm 3 Second stage of the randomized construction phase

Require: Bidirectional connected graph $H = (V, E)$, parameter $\alpha \in [0, 1]$

Ensure: Biconnected graph $G = (V, B)$ with $E \subseteq B$

- 1: $B \leftarrow E$;
 - 2: $p_u \leftarrow \max_{v \in V} \{g(u, v) : [u, v] \in B\}$, for all $u \in V$;
 - 3: **while** G is not biconnected **do**
 - 4: **for all** $u \in V$ that is not an articulation point **do**
 - 5: $g'(u) = \min_{v \in V} \{g(u, v) : u \neq v, v \text{ is not an articulation point and does not belong to the same component as } u\}$;
 - 6: $\underline{g}' = \min_{u \in V} \{g'(u) : u \text{ is not an articulation point}\}$;
 - 7: $\overline{g}' = \max_{u \in V} \{g'(u) : u \text{ is not an articulation point}\}$;
 - 8: Randomly select node $u \in V$ that is not an articulation point such that $g'(u) \leq \underline{g}' + \alpha(\overline{g}' - \underline{g}')$;
 - 9: $B \leftarrow B \cup \{[u, v]\}$, with $v \in V : g'(u) = g(u, v)$;
 - 10: Update the power assignments p_u and p_v ;
 - 11: **return** $G = (V, B)$;
-

- ▶ $P_i = [p_i^1, \dots, p_i^{\phi(i)}]$ is a list of increasing power levels that can be assigned to node $i \in V$.
- ▶ Basic operations applied to each node $i \in V$ operating at the power level p_i^ℓ :
 - ▶ Decrease its current power assignment from p_i^ℓ to $p_i^{\ell'}$, where ℓ' is the highest level which supports a bidirectional edge: total power is decreased by $p_i^\ell - p_i^{\ell'}$.
 - ▶ Increase its current power assignment from p_i^ℓ to $p_i^{\ell+1}$: total power is increased by at least $p_i^{\ell+1} - p_i^\ell$.

- ▶ Local search explores the neighborhood of the current solution, attempting to reduce the total power consumption.
- ▶ A move starts by decreasing the power assignment of one single node, followed by the increase of the power assignments of as many nodes as needed to reestablish biconnectivity.
- ▶ First improving move is accepted and the search moves to the new solution.
- ▶ Procedure stops when no further improving moves exist.

- ▶ Number of power increase operations investigated may be reduced to speedup the local search.
- ▶ Whenever biconnectivity is destroyed by a power decrease, the biconnected components are computed and two acceleration schemes are implemented:
 1. *Reduced scheme*: restricts power increases to pair of nodes belonging to the same biconnected components of the pair of nodes affected by the previous decrease.
 2. *Extended scheme*: considers power increases involving any pair of nodes from different biconnected components.
- ▶ Local search procedure first makes use of the reduced scheme until no further improving moves can be found, then continues using the extended scheme.

- 1 Wireless ad hoc networks
 - Definition
 - Constraints
- 2 Problem statement
- 3 System model and related work
 - Variants
- 4 Integer programming formulation
 - Variables
 - Model
- 5 GRASP heuristic
 - Construction phase
 - Local search phase
- 6 **Numerical results**
 - Experimental settings
 - Optimal solutions
 - GRASP approximate solutions
- 7 Concluding remarks

- ▶ Computational experiments carried out on two classes of randomly generated asymmetric test problems with 10 to 800 nodes:
 - ▶ Euclidean instances:
 - ▶ Nodes uniformly distributed in the unit square grid.
 - ▶ Weights $e(u, v) = F \cdot d_{u,v}^2$, with $F \in [0.8, 1.2]$ being a random perturbation generated from a uniform distribution.
 - ▶ Random instances:
 - ▶ Weights $e(u, v)$ randomly generated in $(0, 1]$.
- ▶ 15 test instances for each problem size and class.
- ▶ Intel Core 2 Quad machine with a 2.40 GHz clock and 8 Gbytes of RAM memory running under GNU/Linux 2.6.24.
- ▶ CPLEX 11.0 was the integer programming solver.
- ▶ Heuristic implemented in C++ using GNU g++ version 4.1 as the compiler, with optimization parameter `-O2`.

Optimal solutions

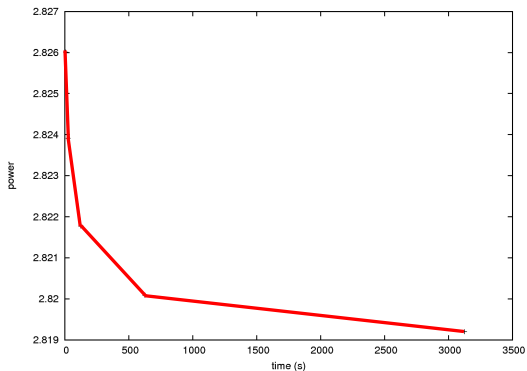
Asymmetric							
Instance	V	unidirectional			bidirectional		
		solved	time (s)	gap (%)	solved	time (s)	gap (%)
Euclidean	10	15	0.89	11.06	15	0.47	7.51
	15	15	16.20	13.75	15	7.55	10.34
	20	15	177.59	13.40	15	66.61	8.10
	25	15	1563.94	11.96	15	298.53	7.71
	30	5	2837.09	7.47	12	1351.98	4.56
	50	–	–	–	–	–	–
Random	10	15	0.07	1.19	15	0.48	5.98
	15	15	0.16	0.00	15	6.99	10.83
	20	15	0.87	0.01	15	117.36	10.87
	25	15	2.36	0.01	15	872.44	13.48
	30	15	5.69	0.02	1	5559.86	13.55
	50	15	126.89	0.02	0	–	–
Symmetric							
Instance	V	unidirectional			bidirectional		
		solved	time (s)	gap (%)	solved	time (s)	gap (%)
Euclidean	10	15	0.78	10.90	15	0.48	7.25
	15	15	16.03	14.23	15	7.24	10.14
	20	15	179.02	12.80	15	47.26	8.27
	25	15	1600.28	12.15	15	509.83	7.70
	30	6	4875.97	11.51	12	1373.72	4.20
	50	–	–	–	–	–	–
Random	10	15	0.11	1.56	15	0.15	0.82
	15	15	0.74	0.40	15	0.23	0.22
	20	15	6.78	0.29	15	2.69	0.28
	25	15	20.43	0.32	15	10.95	0.12
	30	15	102.12	0.22	15	73.71	0.24
	50	12	2827.35	0.07	11	562.42	0.06

- ▶ Minimum power consumption problem is hard to solve.
- ▶ Computation times increase very fast with $|V|$.
- ▶ CPLEX could not solve to optimality in three hours of computation even moderately-sized networks with 30 nodes.
- ▶ Large duality gaps.

- ▶ GRASP heuristic found the optimal solutions for all problems with up to 25 nodes.
- ▶ Computation times to find the known optimal solutions with $|V| = 25$:
 - ▶ Euclidean instances solved in less than one second.
 - ▶ Random instances were harder and took approximately 20 seconds on average.
- ▶ Average objective values over five runs (one instance) as the running time limit increases from five to 3125 seconds:

		$ V = 200$				
Instances	5 s	25 s	125 s	625 s	3125 s	
Euclidean	1.74172	1.73922	1.73778	1.73714	1.73640	
Random	17.72963	17.64030	17.58805	17.50900	17.46120	
		$ V = 400$				
Instances	5 s	25 s	125 s	625 s	3125 s	
Euclidean	2.82601	2.82391	2.82177	2.82009	2.81920	
Random	25.08528	24.91952	24.83406	24.74227	24.68739	

- ▶ Continuous improvement in solution quality along the computation time for one Euclidean instance with $|V| = 400$:



GRASP results: comparison (power consumption)

- ▶ GRASP vs. MST-Augmentation heuristic of Calinescu and Wan, 2006:
 - ▶ GRASP: one hour of computation time for each instance.
 - ▶ MST-Augmentation is faster: a few seconds for $|V| = 800$.
 - ▶ GRASP finds much better solutions.

Instances $ V $	MST-Augmentation		GRASP		gain (%)	
	power (max.)	total power	power (max.)	total power		
Euclidean	25	0.18784	2.18646	0.15052	1.35089	38.22
	50	0.11935	2.04944	0.07861	1.27900	37.59
	100	0.06809	2.15343	0.04066	1.32409	38.51
	200	0.03857	2.84844	0.02249	1.76355	38.09
	400	0.03199	4.72278	0.01394	2.82529	40.18
	800	0.02999	8.44331	0.01057	4.99252	40.87
Random	25	0.92757	12.84646	0.55168	5.46654	57.45
	50	0.95827	24.20324	0.48981	8.35444	65.48
	100	0.97475	44.97218	0.41067	11.86145	73.62
	200	0.99056	85.08285	0.31773	17.14185	79.85
	400	0.99039	158.26410	0.23502	25.00776	84.20
	800	0.99648	293.63736	0.18951	37.24512	87.32

GRASP results: comparison (graph structure)

Instances	V	MST-Augmentation			GRASP		
		arcs (avg.)	edges (avg.)	degree (avg.)	arcs (avg.)	edges (avg.)	degree (avg.)
Euclidean	25	130.00	54.73	4.37	81.60	32.80	2.62
	50	275.73	111.60	4.46	168.33	65.66	2.62
	100	608.73	239.20	4.78	327.66	130.13	2.60
	200	1481.66	556.53	5.56	681.00	254.73	2.54
	400	4502.13	1524.40	7.62	1538.20	507.93	2.53
	800	14707.06	4530.40	11.32	3705.60	1033.06	2.58
Random	25	329.33	100.60	8.04	149.93	31.26	2.50
	50	1224.93	332.13	13.28	432.80	61.40	2.45
	100	4509.73	1083.60	21.67	1197.80	131.20	2.62
	200	16972.53	3748.93	37.48	3385.40	264.53	2.64
	400	63143.00	12738.66	63.69	9583.46	532.73	2.66
	800	233383.66	43080.53	107.70	27377.60	1077.20	2.69

- ▶ GRASP solutions have fewer arcs/edges and smaller power assignments (useful to mitigate interference).
- ▶ Average node degrees show that GRASP solutions are very close to the theoretical lower bounds.

- 1 Wireless ad hoc networks
 - Definition
 - Constraints
- 2 Problem statement
- 3 System model and related work
 - Variants
- 4 Integer programming formulation
 - Variables
 - Model
- 5 GRASP heuristic
 - Construction phase
 - Local search phase
- 6 Numerical results
 - Experimental settings
 - Optimal solutions
 - GRASP approximate solutions
- 7 Concluding remarks

Concluding remarks

- ▶ Integer programming formulation for the bidirectional topology variant of the biconnected minimum power consumption problem.
- ▶ Formulation can be easily extended to problems with other connectivity requirements.
- ▶ Formulation applied to four variants of the problem:
 - ▶ Symmetric or asymmetric input graphs
 - ▶ Unidirectional or bidirectional solutions
- ▶ State-of-the-art integer programming solver could not solve large instances to optimality.

- ▶ GRASP heuristic proposed to find good approximate solutions for real-size problems.
- ▶ Heuristic applied to large instances of the asymmetric input with bidirectional topology variant.
- ▶ Experimental results for large networks with up to 800 nodes showed that:
 - ▶ GRASP heuristic is relatively fast.
 - ▶ GRASP heuristic finds effective solutions which significantly improved those obtained by a literature heuristic.
 - ▶ Solutions obtained by the GRASP heuristic are very close to the optimal solutions.