Telecom Applications of OR

Tami Carpenter
Telcordia Technologies

Why does the phone company have operations researchers?

Choices:

a) We are good at remembering acronyms

b) Habit

c) To improve their bottom line

d) All of the above

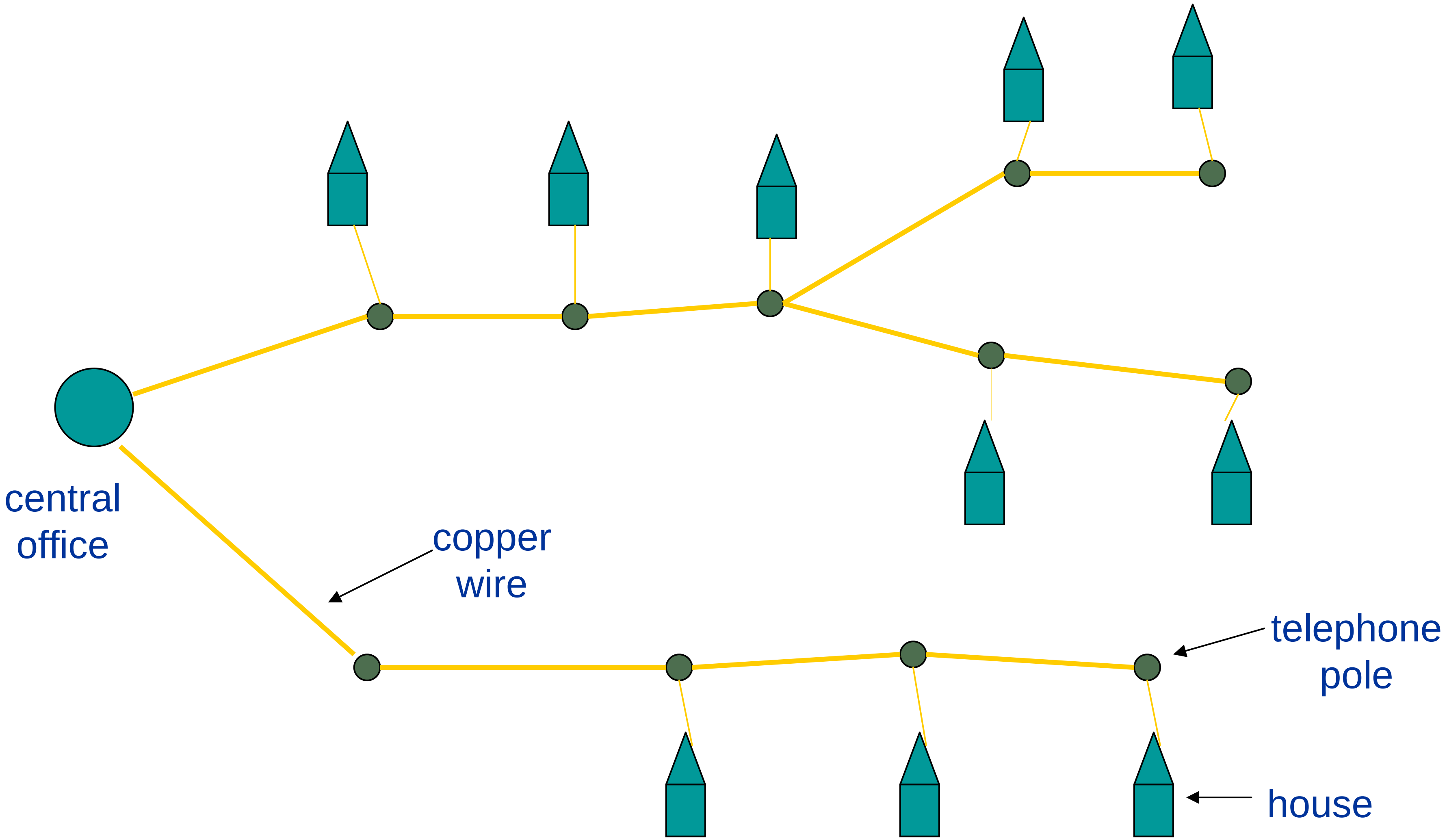
What types of problems do we work on?

- Network Design & Planning
  - What equipment do we put in the network & where?
  - How much capacity do we need?
- Network Provisioning
  - How do we allocate network resources?
  - How do we route traffic in the network?
- Network Analysis
  - How reliable is the network?
- Network Security
  - How do you detect network attacks?

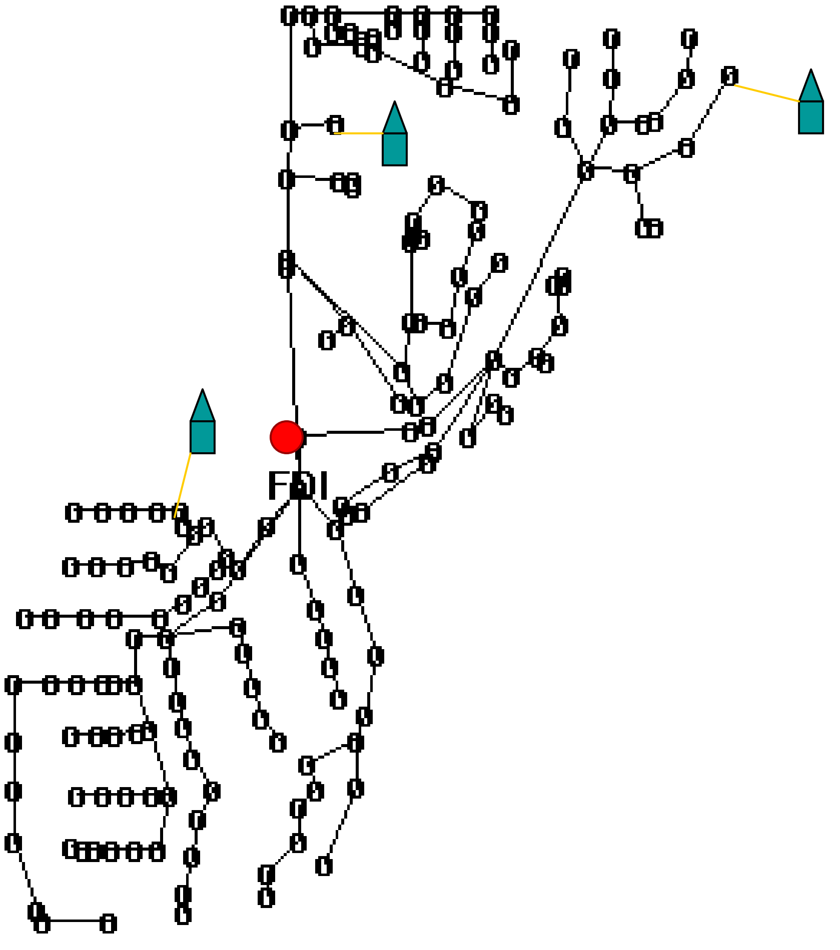
Planning DSL and Beyond: One of my favorite problems

- What would you rather have high speed or dial up?
  - More IS better…
- Before the mid 90s, it was ALL dial-up
- How could the telephone companies upgrade their network to compete with cable?
  - DSL
  - Fiber-to-the-curb
  - Other newer choices

Access network anatomy

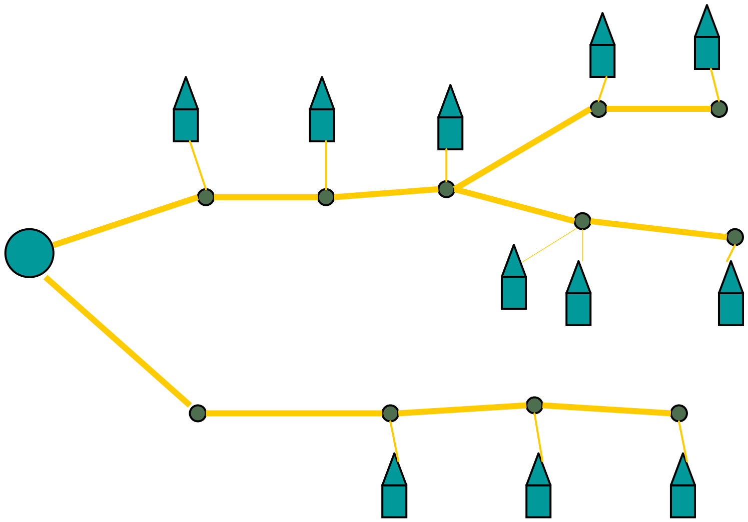


A real access network in Iowa



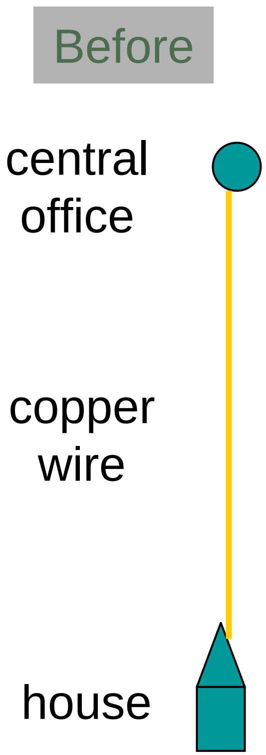
Key features of the access network

- Logically, it looks like a TREE
  - It is “rooted” the central office
  - The houses are the leaves
  - Intermediate nodes are poles

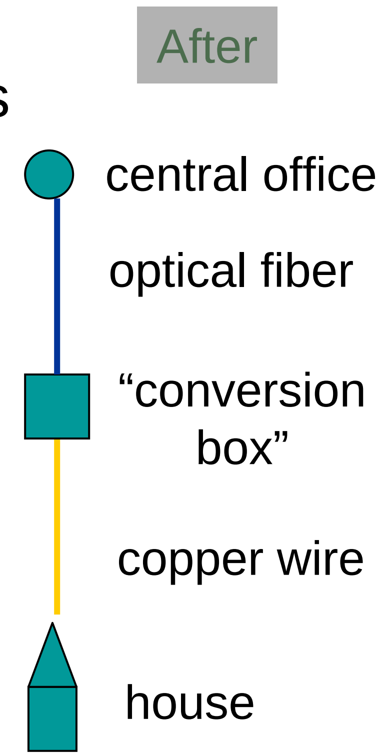


Issue: some of the houses are too far from the CO to get broadband service

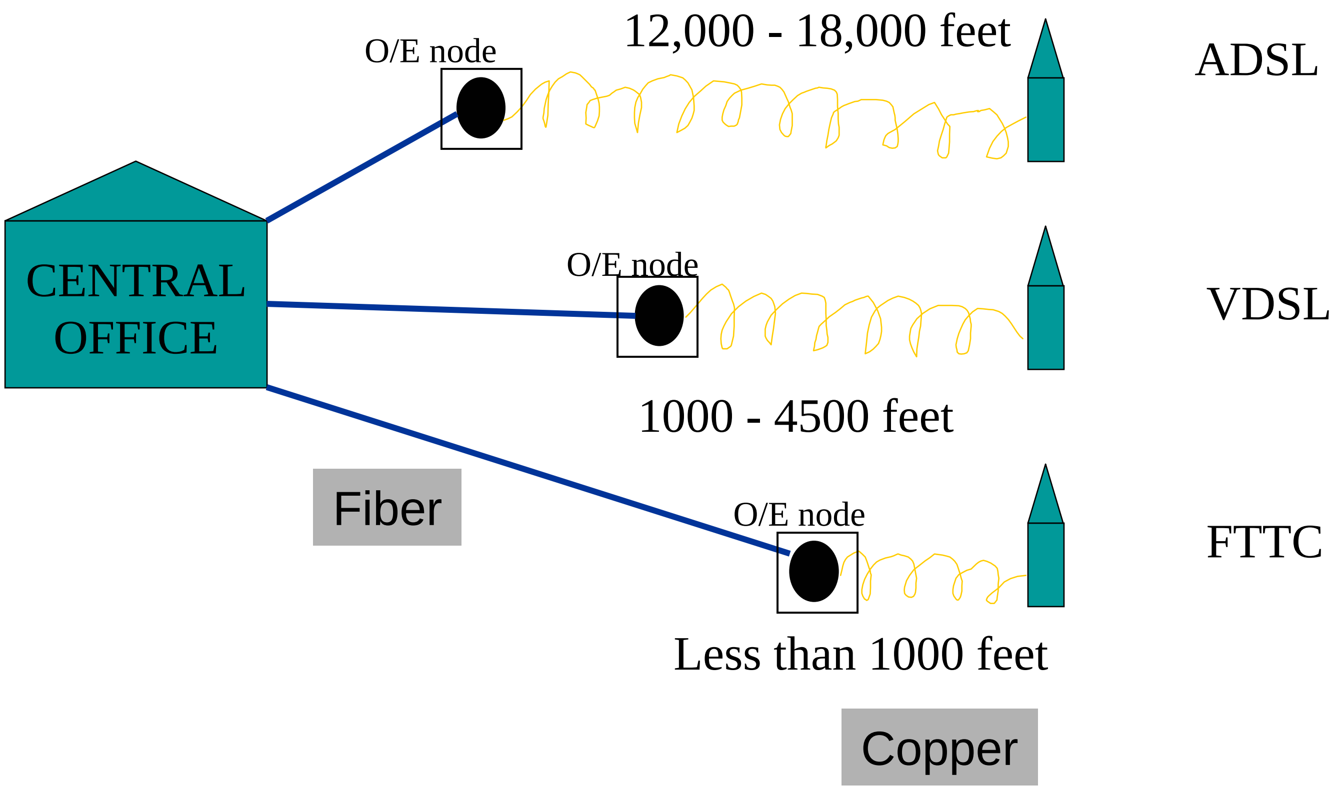
Access network evolution



- Before 1990's:
  - Copper wires carry electrical signals
  - Signal degrades with distance
  - Can't get enough through!
- Fiber optics:
  - Huge amount of information
  - Over long distances
  - With little degradation
- Broadband access networks use fiber optic cables near the CO

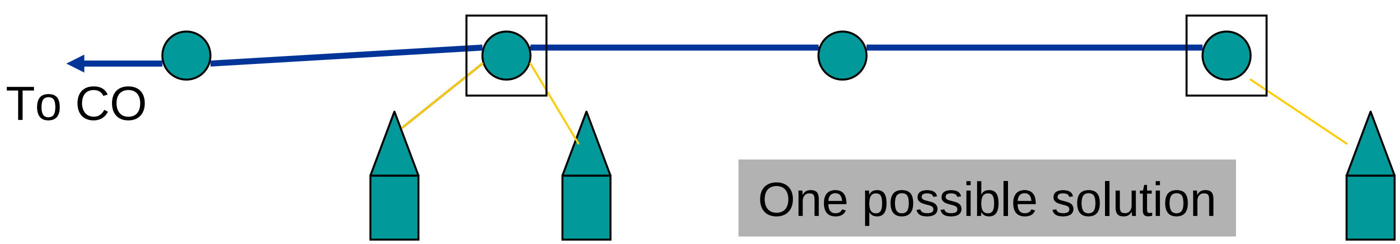


Sample of broadband “flavors”



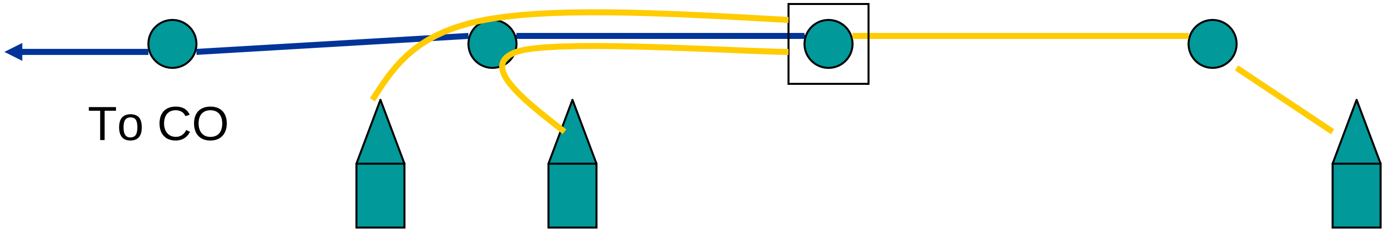
Installing Fiber

- When a house is too far, we install fiber part of the way from the CO
- When we install fiber, we need a "conversion box" to translate between electrical & optical
- Conversion boxes are placed at telephone poles
- There is a limit on allowable distance between house and conversion box



A better solution

- Conversion boxes are expensive, so we want to use as few as possible



Better, if it's legal

The constraints

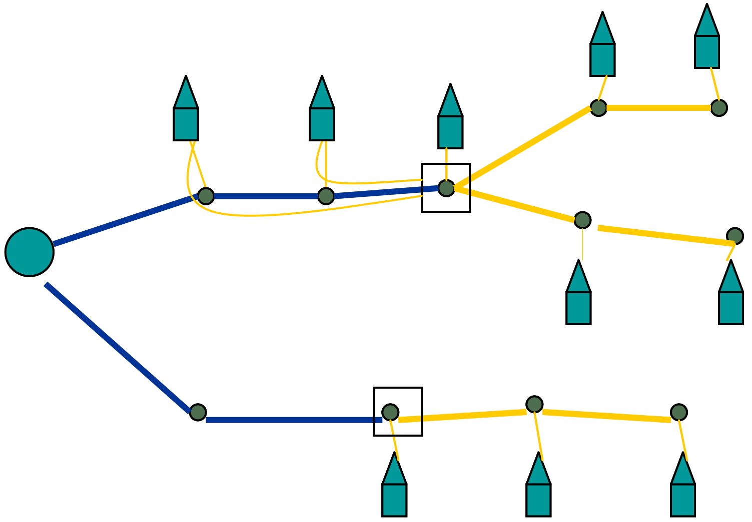
- A house must be within some prescribed distance of it's CB
  - Distance is along the tree
- A CB can serve only a prescribed number of houses
  - We'll ignore this for a while…

The Problem (Version 1)

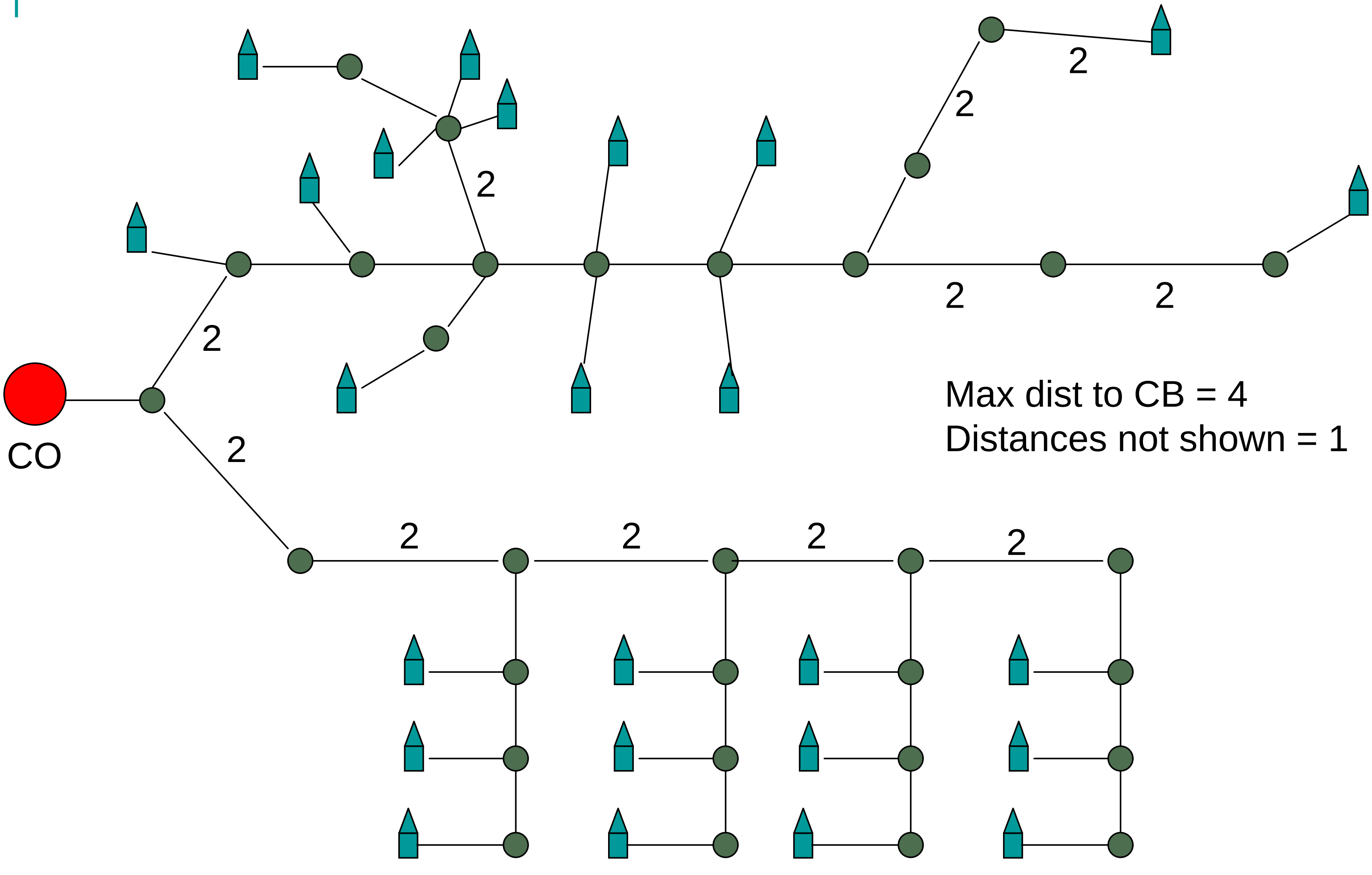
- Given a tree whose links have associated distances, place CBs at the nodes of this tree and assign each leaf (house) to a CB so that no leaf is more than distance *D* from its assigned CB and the number of CBs is minimized.
  - Assign each house to a CB that is within the maximum allowable distance (feasibility)
  - Minimize the number of CBs

A small example

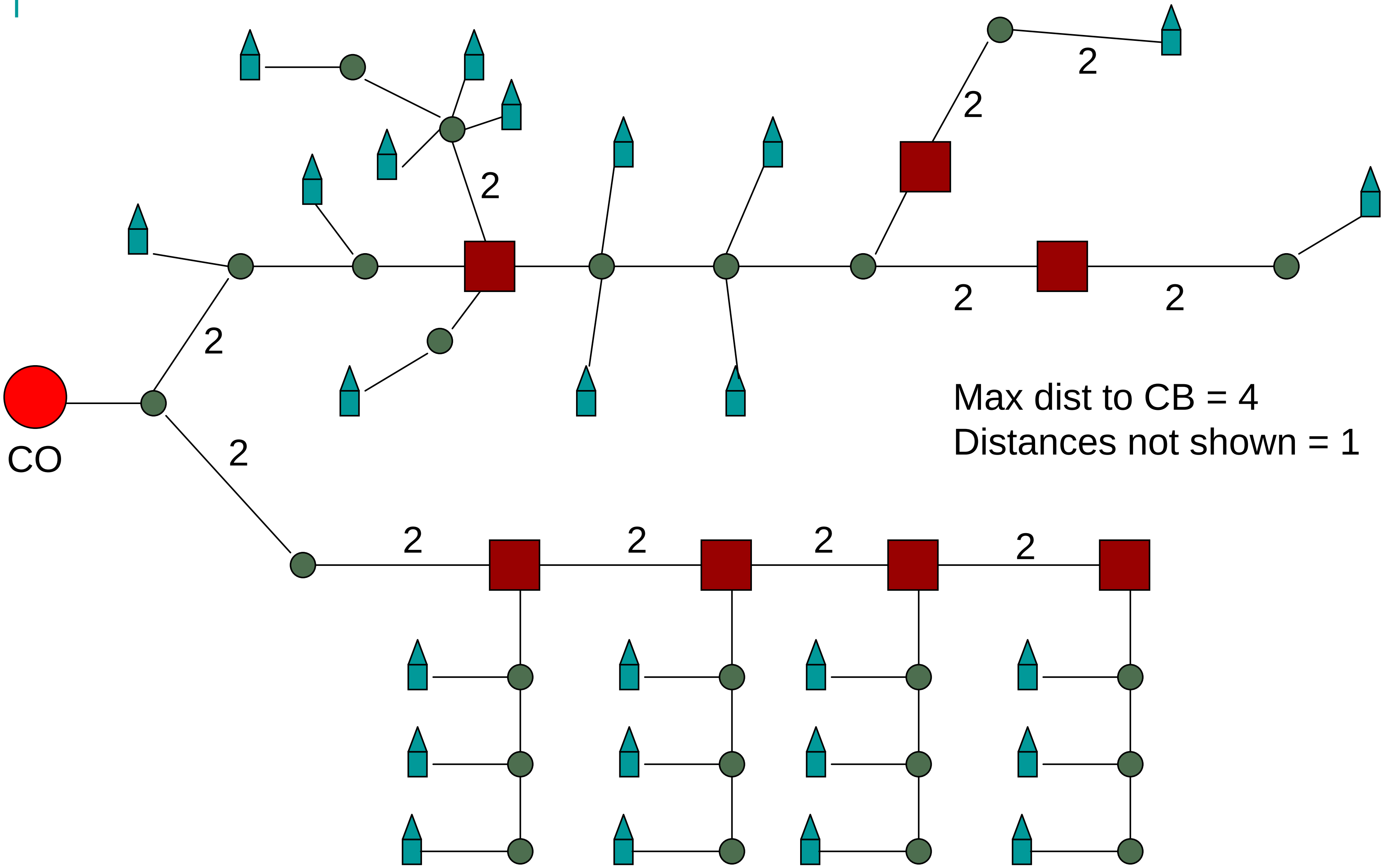
- Each link is 1 unit long
- A house can be no more than 3 units from its CB



An exercise



A solution



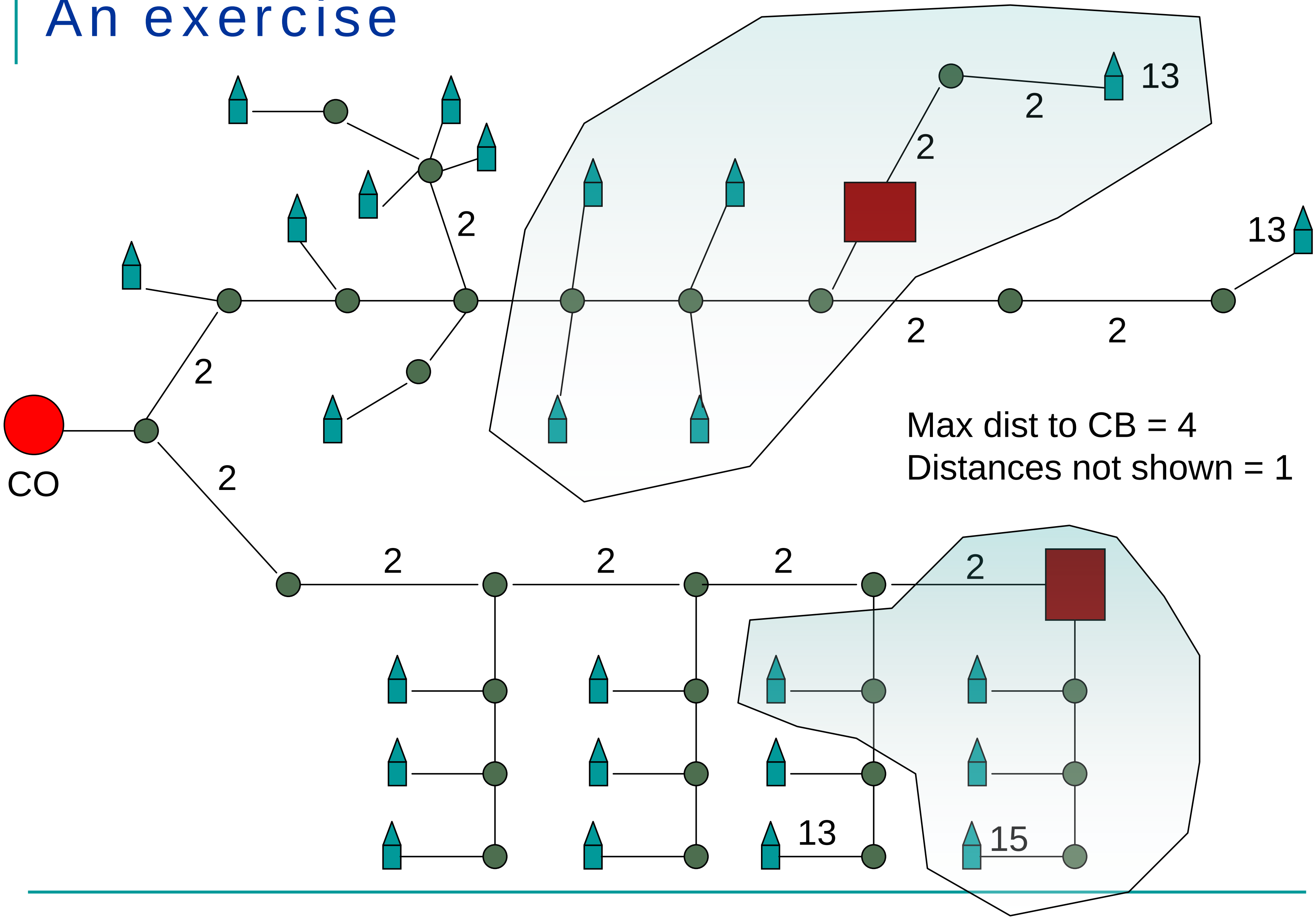
Solution uses 7 CBs

An optimal algorithm

- Sort the houses so that they are in order of decreasing distance to the root
- Place a CB as close to the root as possible, given that it can still be reached by the first house on the list
- Attach any house within the allowable distance to be served by this CB and remove it from the list

Iterate

An exercise


CO
2
2
2
2
13
2
2
13
2
2
Max dist to CB = 4
Distances not shown = 1
2
2
2
2
13
15

An integer programming digression

- Would we rather formulate and solve an integer program?

x_{ij} is 1 if house *i* is assigned to a CB at pole *j*

y_j is the number of CBs placed at node *j*

Minimize: $\sum_j y_j$

s.t. $\sum_{j:d(i,j)\leq D} x_{ij} = 1 \quad \forall i$ — Every house is assigned

$\sum_i x_{ij} \leq C y_j \quad \forall j$ — Sufficient capacity at each pole

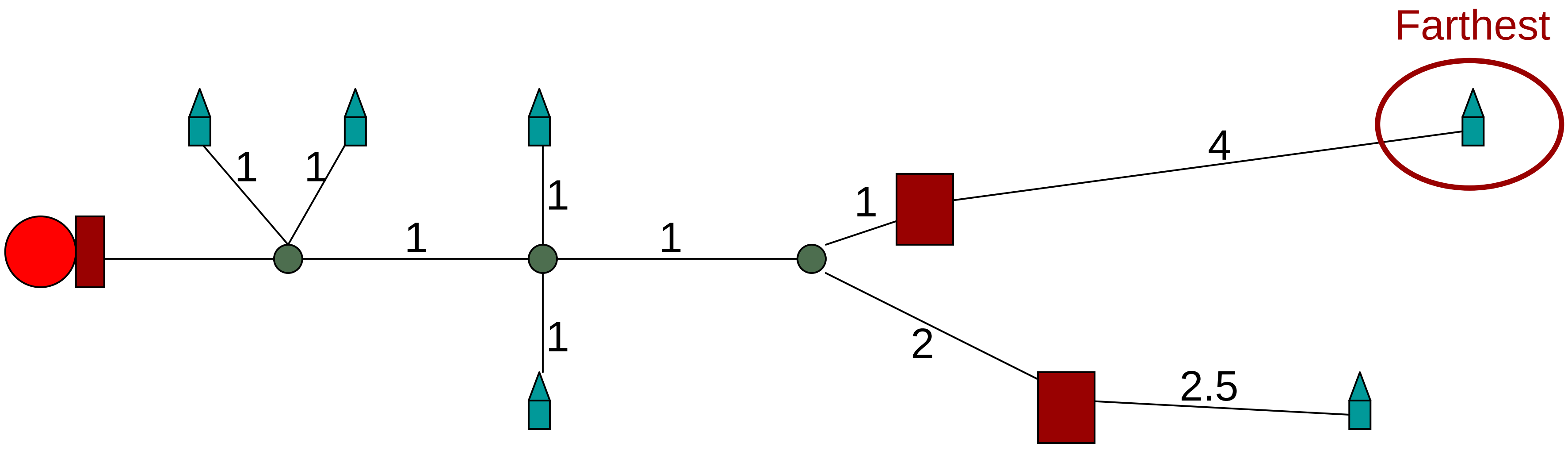
y_j integer $\forall j$

$x_{ij} \in \{0,1\} \quad \forall i, j$

The Problem (Version 2)

- What about when CBs have fixed capacity?
  - Place CBs at poles so as to
    - Minimize the total number of CBs
    - Assure that every house is assigned to a CB that is close enough
    - AND assure that no CB's capacity is exceeded
- Does our algorithm still work?

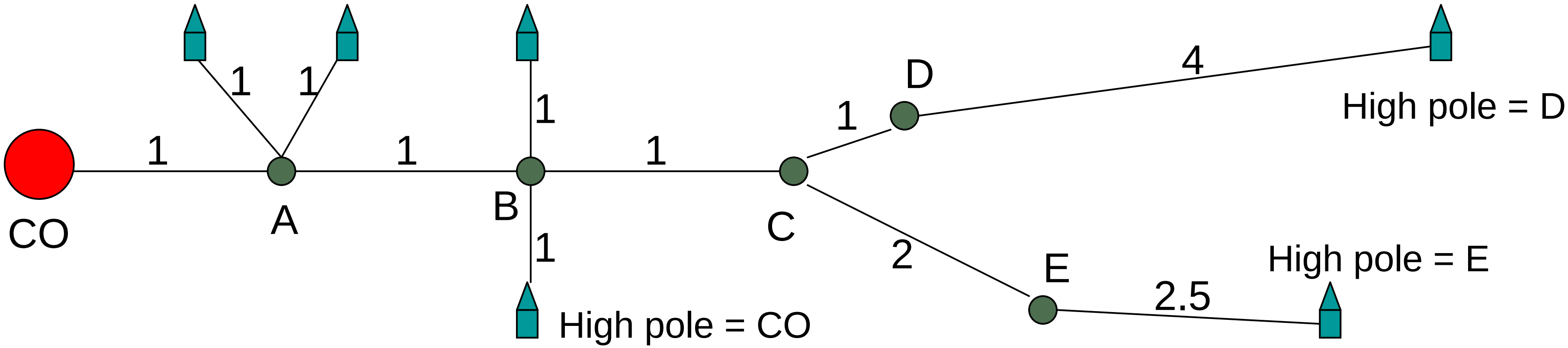
An example where it doesn't work…



- CB capacity = 3
- Distance limit = 4
- Where does the first CB go?

A definition

- Associate with each house a “high pole”, which is the closest pole to the root that this house could reach
- This is as close to the root as you could place a CB to serve this house

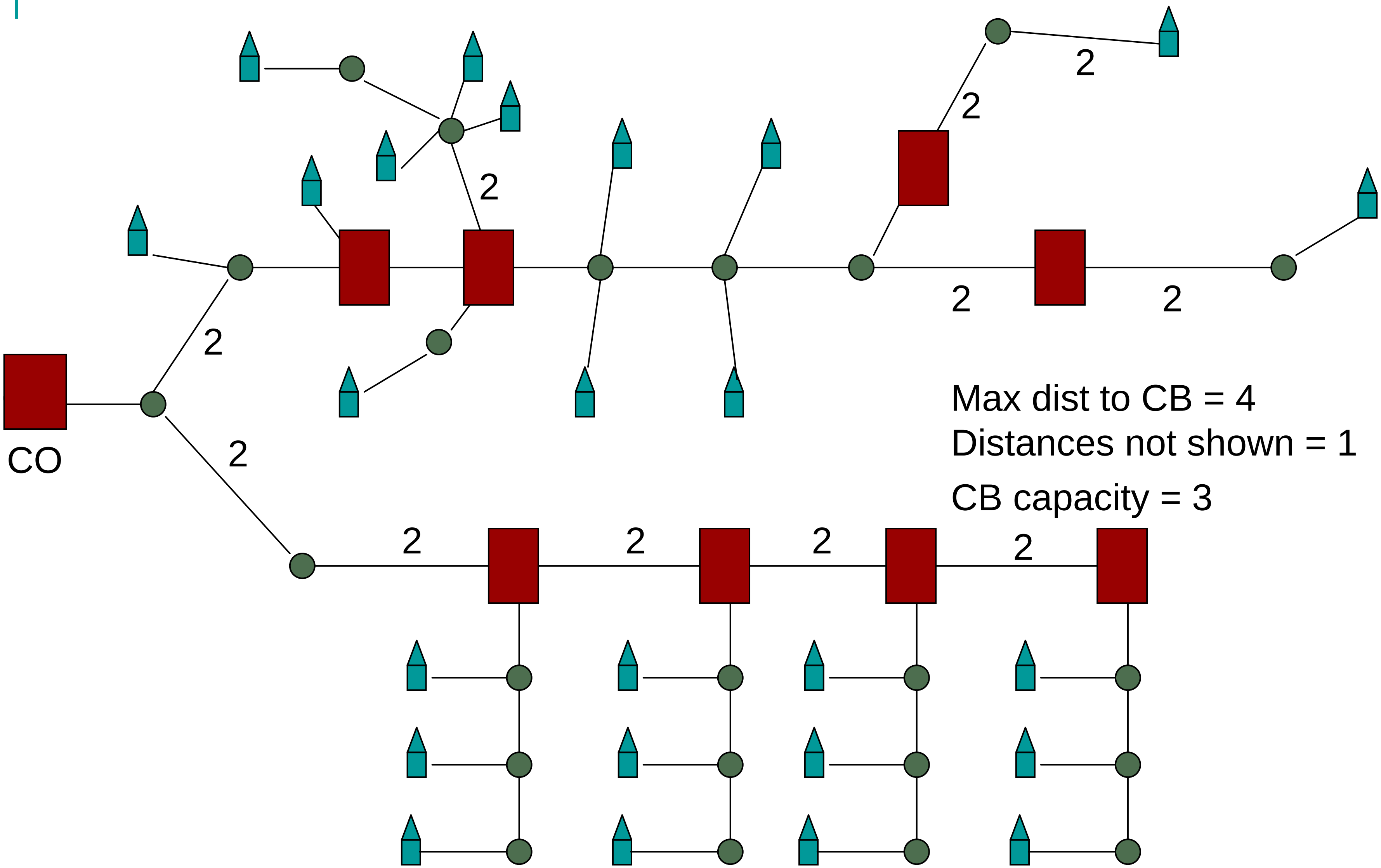


The new algorithm

- Sort the houses so that they are in order of decreasing high-pole-to-root distance.
  - If there's a tie, put the houses closer to the root lower in the list
- Place a CB at the high-pole of the first house on the list
- Attach the first *C* houses that can be served by this CB and delete them from the list

Iterate

An exercise



Max dist to CB = 4

Distances not shown = 1

CB capacity = 3

A few remarks

- This is an optimal algorithm
  - Paper by Jaeger & Goldberg, 1994
- It is a “greedy” algorithm
  - We only add a CB when we have to
- We start far from the leaves and work our way in
  - Does the other way make sense?
  - Does it make any difference which node is the root?
- There are a few subtleties…

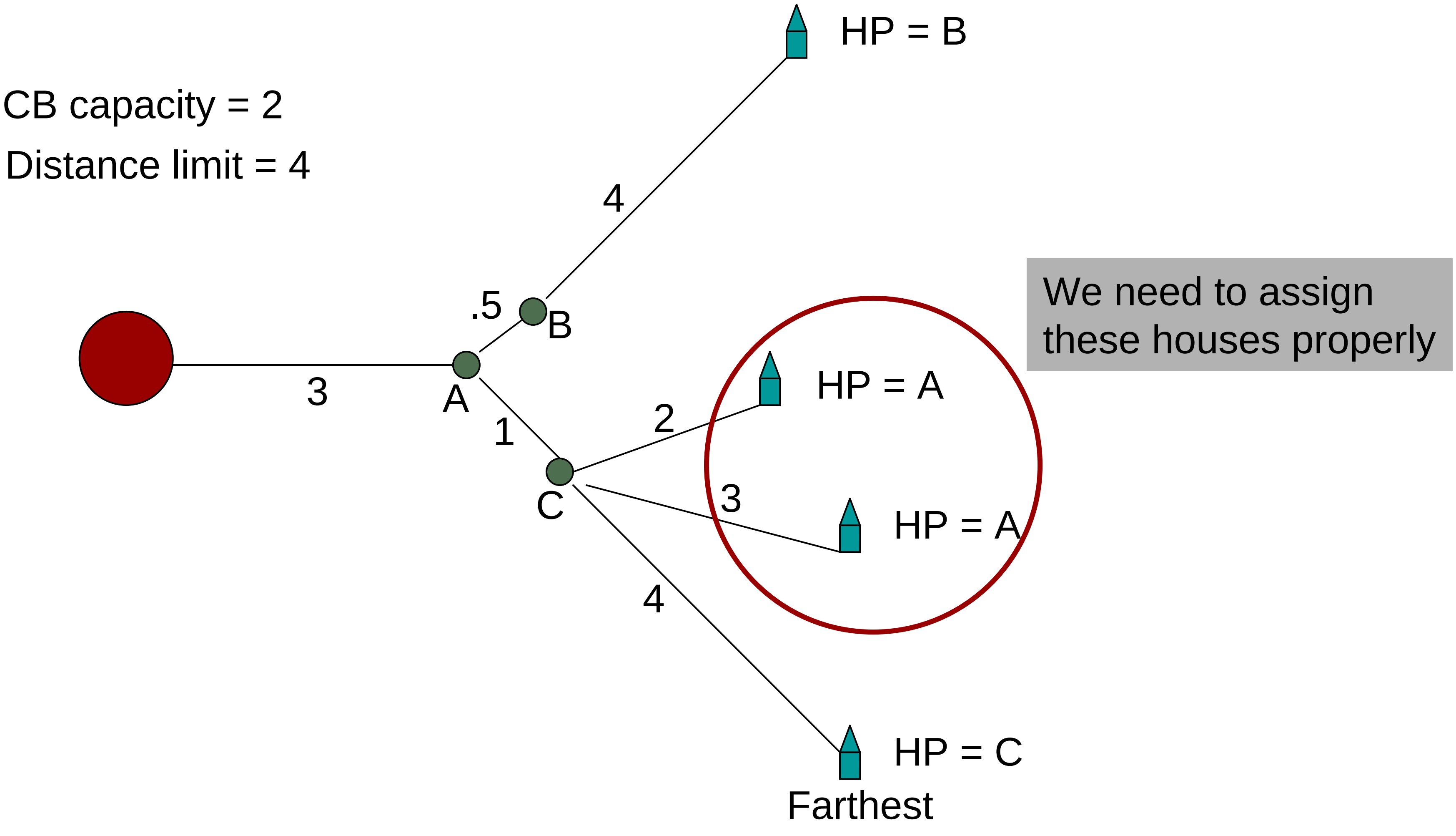
What about this?

- Sort the houses so that they are in order of decreasing high-pole-to-root distance.
  - If there’s a tie, put the houses closer to the root lower in the list
- Place a CB at the high-pole of the first house on the list
- Attach the first N houses that can be served by this CB and delete them from the list

Iterate

Can we just do everything in terms of high-pole distance?

An example



HP = B
CB capacity = 2
Distance limit = 4
4
.5
B
3
A
1
C
2
3
4
HP = A
HP = A
HP = C
Farthest
We need to assign these houses properly

Some variations on this problem

- What about when there are several different CB sizes available?
  - This appears hard…subject of Mazur's PhD thesis
  - This is an important part of the "real problem"
- What about when you can place CBs at only certain poles?
- What about when some CBs already exist?
- What about when the graph is not a tree?
  - Uncapacitated version is a classic "hard" problem (dominating set)

The "real" problem

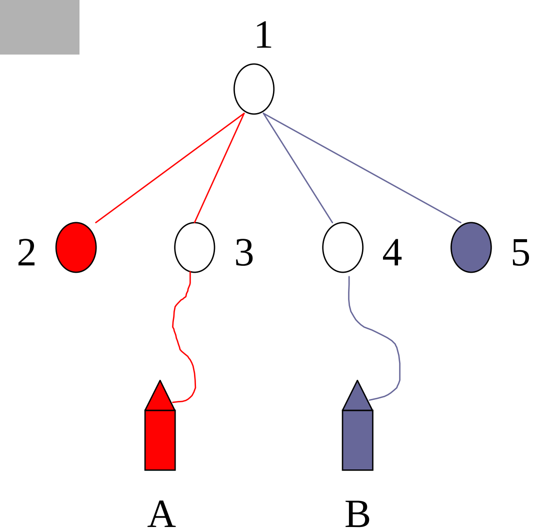
- Two key additions:
  - There are multiple different sized CBs for different costs
    - The choice of CB size is now part of the problem
    - Still looking for a fast, optimal algorithm for this!
  - There are additional contiguity constraints to promote more "maintainable" designs
    - Makes the multiple-facility problem tractable using dynamic programming
    - Paper by Carpenter, Eiger, Seymour & Shallcross, 2001

Contiguity constraints

- Assure that if two copper wires “meet” on their way to a CB, they will route to the same CB node

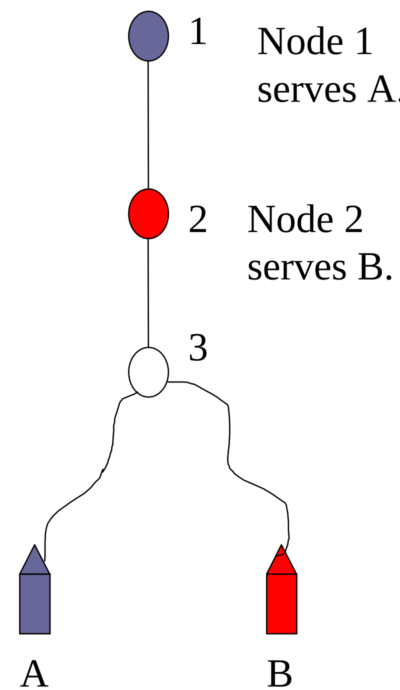
Examples of excluded solutions

Example 1:

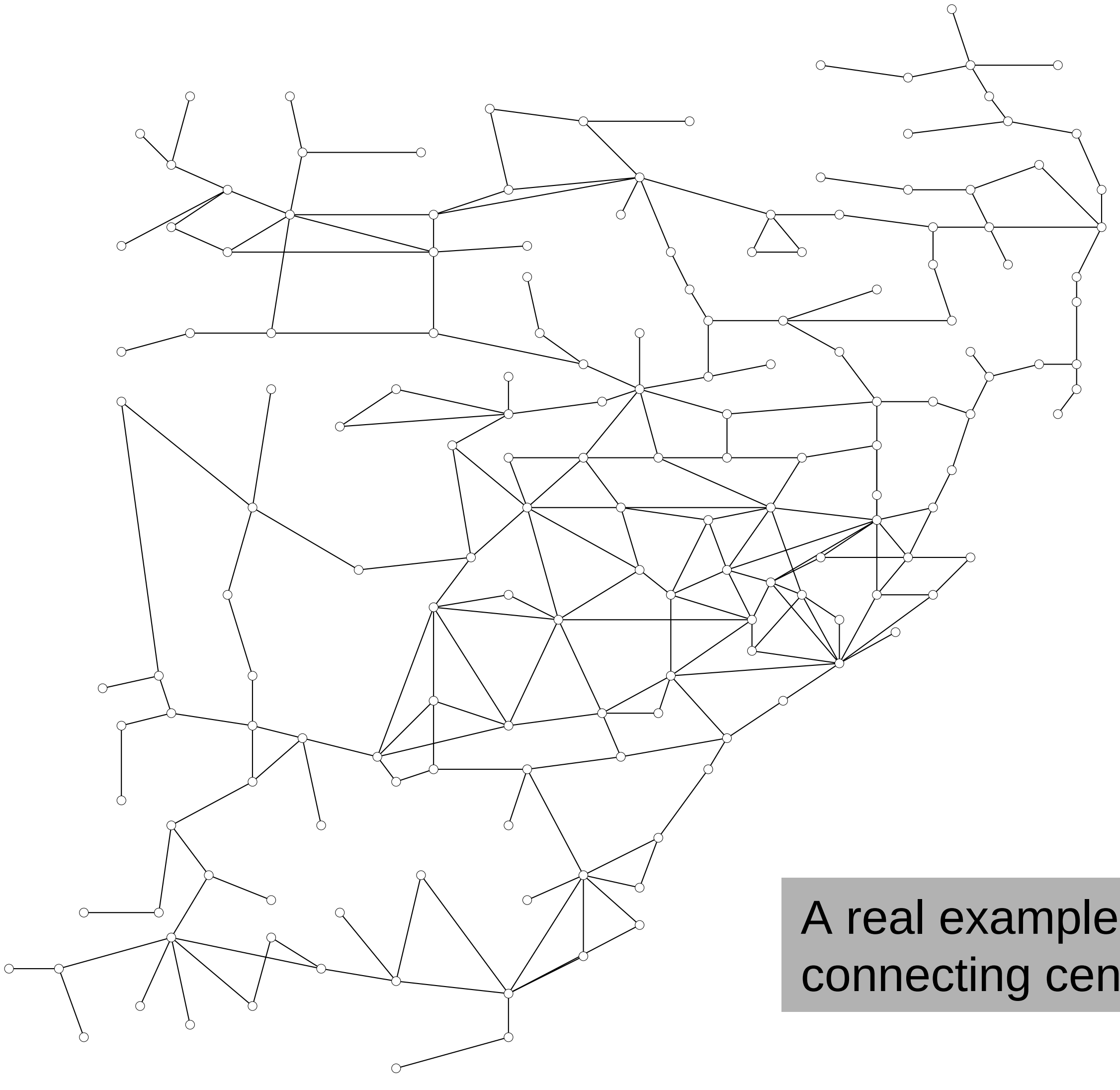


Node 2 serves house A.
Node 5 serves house B.

Example 2:



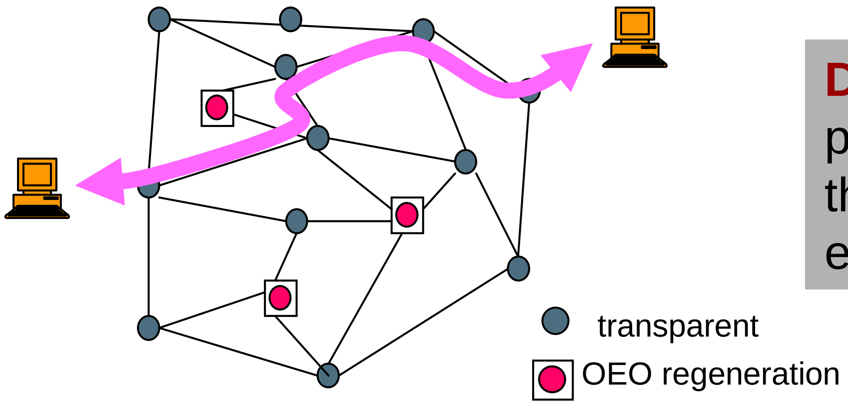
A related problem in the “core” network



A real example of a network
connecting central offices

Designing optical networks

- In today's optical networks, optical-electrical-optical (OEO) signal regeneration occurs at equipment at each node
  - Future networks will switch optically
  - Signal impairments limit feasible optical paths
    - constrain the number of consecutive optical nodes in a path
    - constrain the distance between OEO nodes in a path

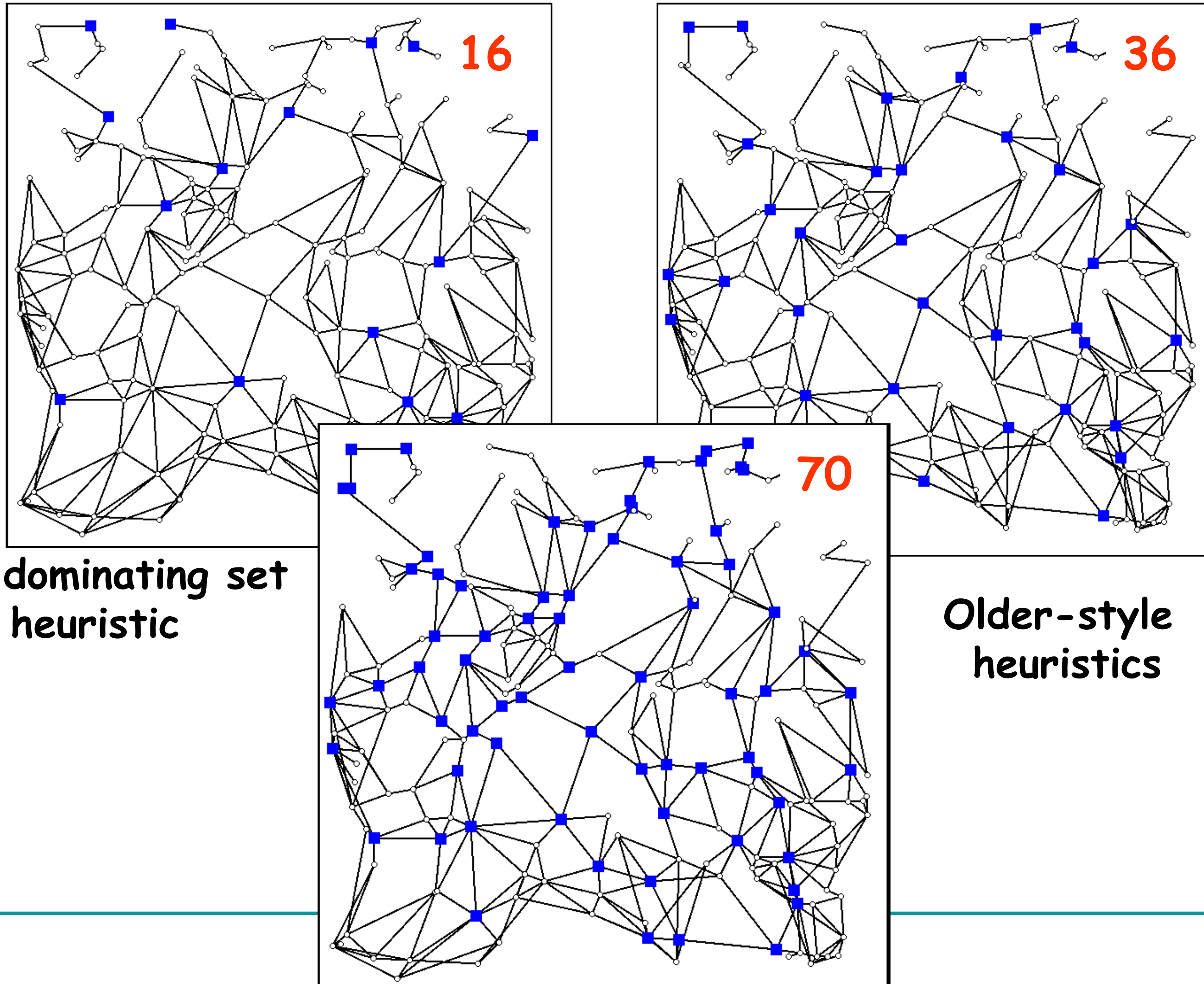


Design problem: where do we provide OEO regeneration to assure that a feasible path exists between each pair of nodes?

Locating OEO at nodes

- Select the smallest set of OEO nodes that provides at least one impairment-feasible path between each pair of nodes
- A variant of the Connected Dominating Set Problem:
  - Create a graph G that has a link between every pair of nodes that can communicate without regeneration
  - Select a set minimum size set of (OEO) nodes such that every node not in the set is connected to a node in it
    - The selected set is a dominating set in graph G
    - This assures that each non-OEO node reaches an OEO node
  - …and such that if we remove all nodes not in the set the remaining graph is connected
    - The selected set is a connected dominating set
    - This assures that OEO nodes can feasibly communicate

A sample solution



New dominating set heuristic

Older-style heuristics