Terrain Analysis in Strategy Games

CS 395GAI

Spring, 2005

Overview

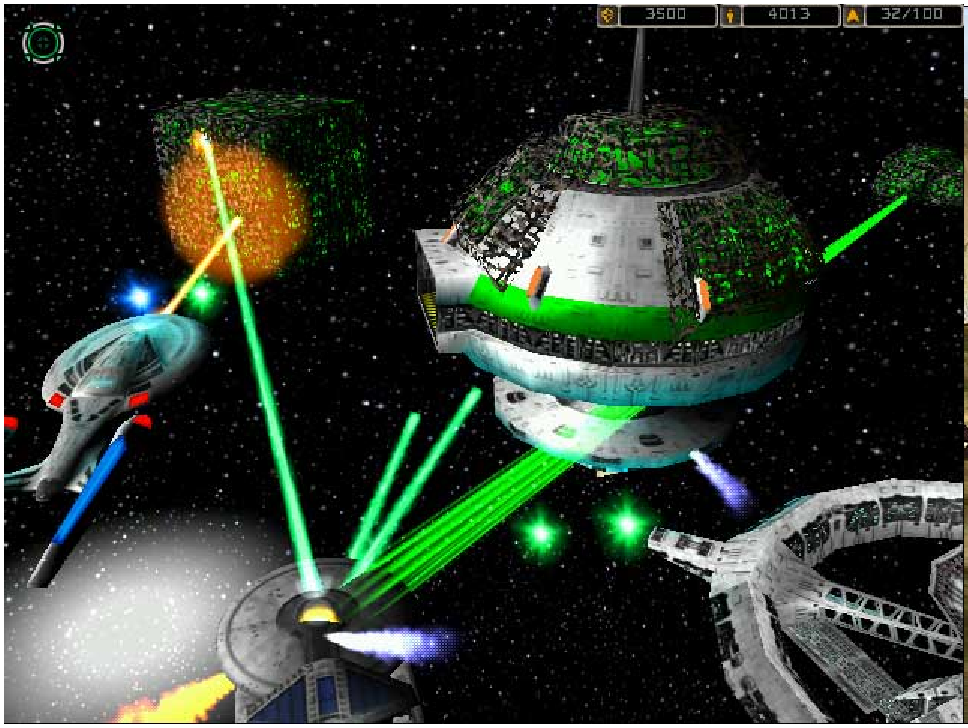
- Review
  - Baseline representations of space in games
  - Path-finding
    - Example: A*
  - Position-finding
    - Example: Influence maps
- Terrain problems in strategy games

From the developer's side

- "Unfortunately, we had to tweak the quality/heuristics of the basic pathfinding algorithm [in AOE1] to make it work effectively for the CP AI canPath checks. As a result, we ended up with a one-size-didn't-quite-fit-all implementation for the pathfinding. To put it nicely, we got roasted for that."
- "*Age of Empires 2* spends roughly 60 to 70% of simulation time doing pathfinding"
  - Dave C. Pottinger, Ensemble Studios, in *Terrain Analysis in Realtime Strategy Games*

Three families of spatial solutions

- Tiles
- Waypoints
- Quad trees



Tiles

- Variations
  - Can be rectangular or isometric, depending on perspective
  - Some games use hexagonal grids to model distances traveled on diagonals more accurately

- Tradeoffs
  - Very simple
  - Uniform resolution can waste storage on uninteresting regions of space

Quad Trees

- Carve up space according to where objects aren’t
- Stop conditions:
  - Uniform contents
  - Maximum depth on recursion reached
- Tradeoffs
  - Provides variable resolution
  - More intricate to generate and use

Waypoints

- Annotate terrain with hand-selected places that movable entities can be in
  - Adjacency relationships between waypoints indicate ways to move from one to the other (including travel time)
  - Additional annotations can be used to indicate other properties
    - whether line-of-sight exists between two waypoints
    - Part of a room or a base or some interesting location
    - Environment conditions, such as light/dark, types of movement required
- Tradeoffs
  - Can analyze to derive many useful tactical properties
  - Generally must be entered by hand

Path-finding

- All three spatial solutions give rise to common formal framework for finding paths
  - Graph search
  - Costs on links of graph
- Results from early AI research universally used in game development
  - E.g., A* search and its successors
- Game developers have invented many improvements
  - Nothing like trying to live within a tight CPU budget to unleash creativity!

A*: Formalization

- Node = a state in your search corresponding to a place in your terrain representation
  - Node contains path to get from start to that place
  - Multiple paths can go through the same place, so there can be more nodes than places
- Children = nodes corresponding to adjacent places in your terrain representation.
- Links between nodes have *costs*.
  - Depends on distance
  - Depends on difficulty of movement
  - Can roll in other factors, e.g. concealment/visibility, to incorporate tactical factors

A* formalization, continued

- Start, Goal = nodes
- g(n) = cost to get to this node from your starting position
  - Sum of costs so far along this path
- h(n) = Estimate of cost remaining to goal
  - Intuition: Path cost estimate = g(n) + h(n)
  - Use estimate to explore cheapest paths first
  - If h never overestimates cost remaining, then h is *admissible*
  - If h is admissible, then A* is guaranteed to be optimal
    - Will always find the cheapest path
    - Will always examine the fewest nodes

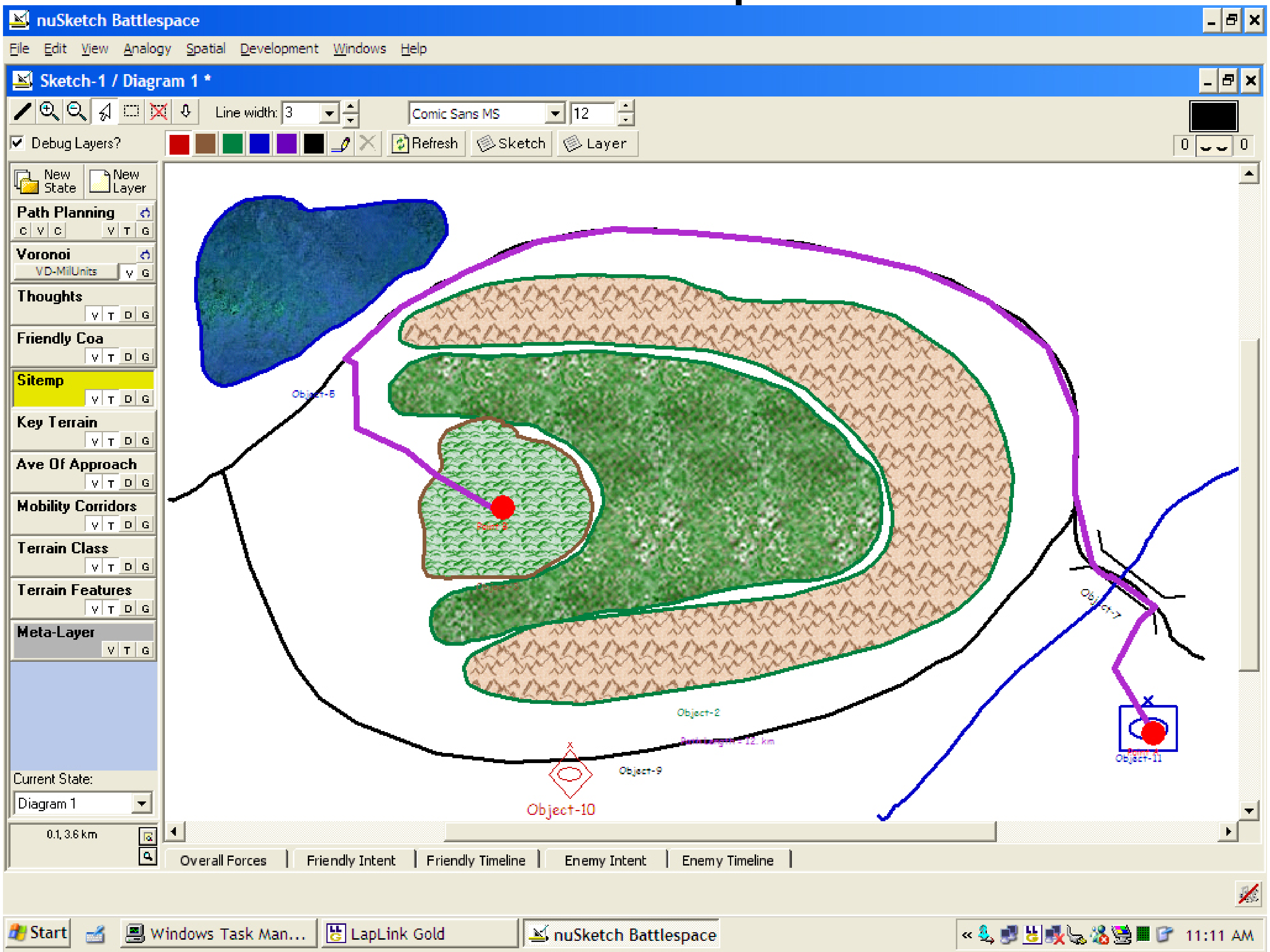
A*

1. Let Open = {make-node(start)}, Closed = {}
2. If Open = {} then return failure
3. Let N = best node from Open
   a) If place(N) = goal then return N
   b) For each child C of N,
      i. Is there a node N2 with place(N2) = place(C) in Open or Closed?
         a. If so, replace path(N2) with path(C) if f(N2) > f(C)
         b. Otherwise, add C to Open
   c) Move N to Closed

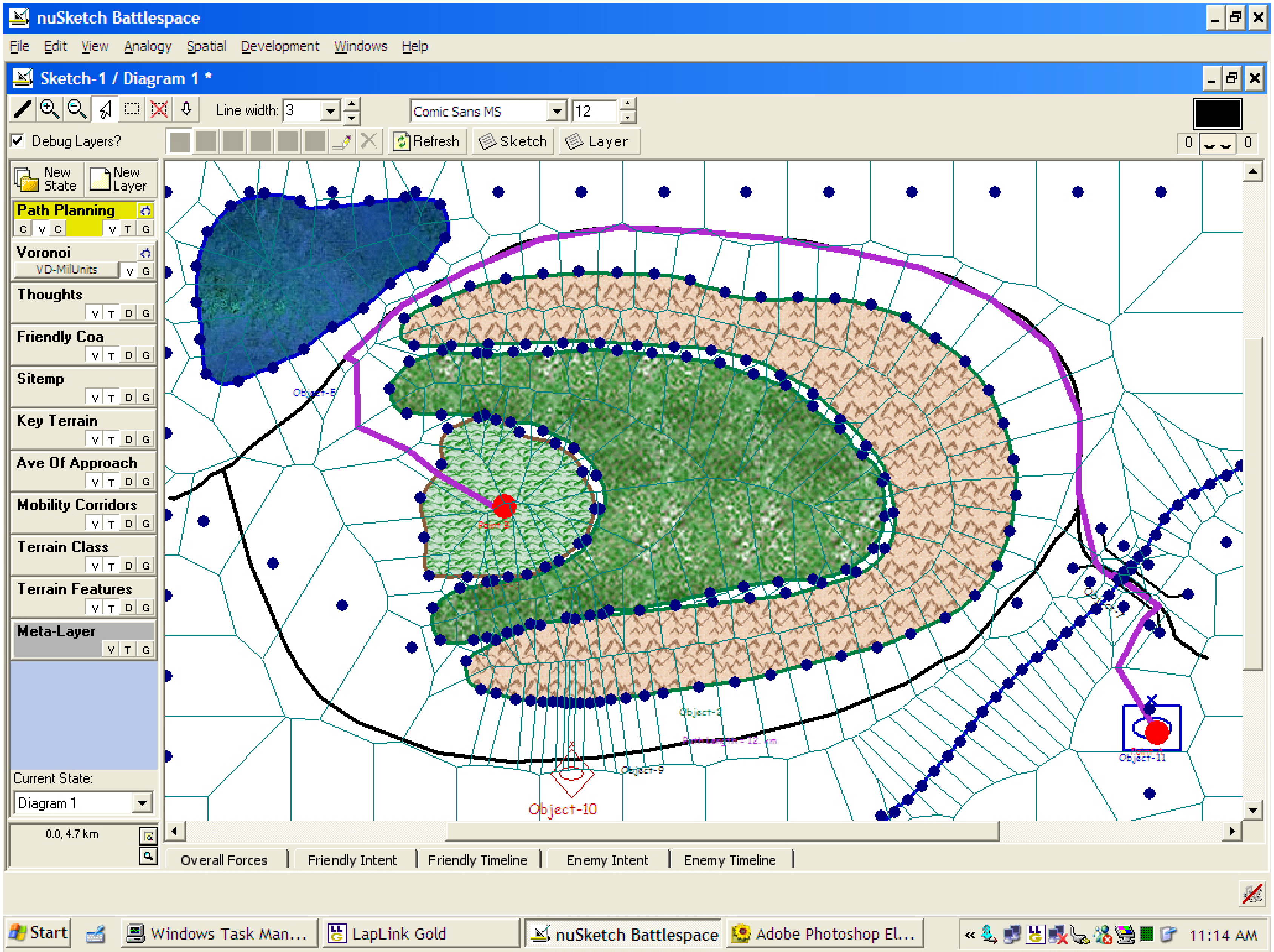
Demo

- http://www.ccg.leeds.ac.uk/james/aStar/

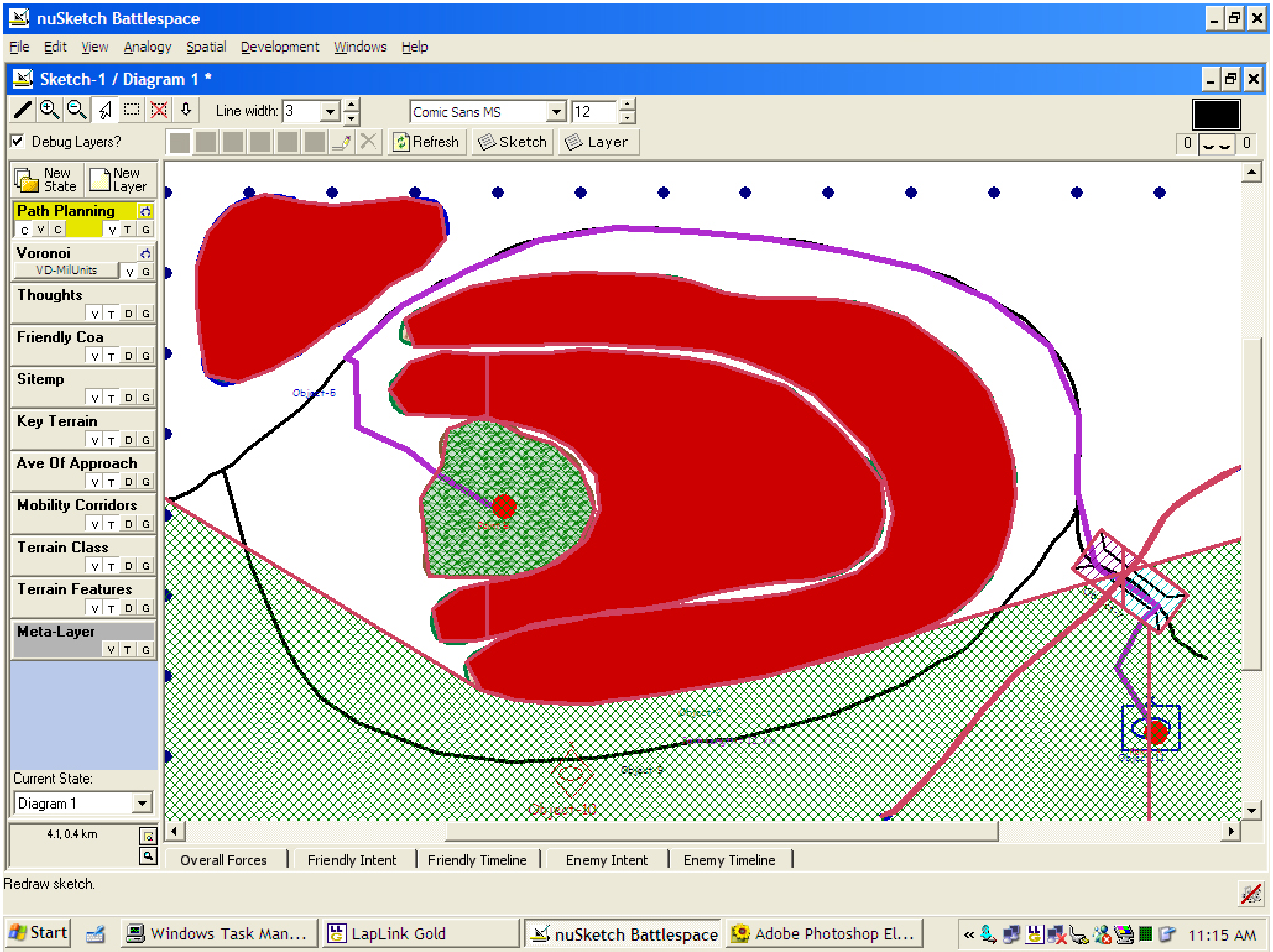
A* can involve multiple constraints



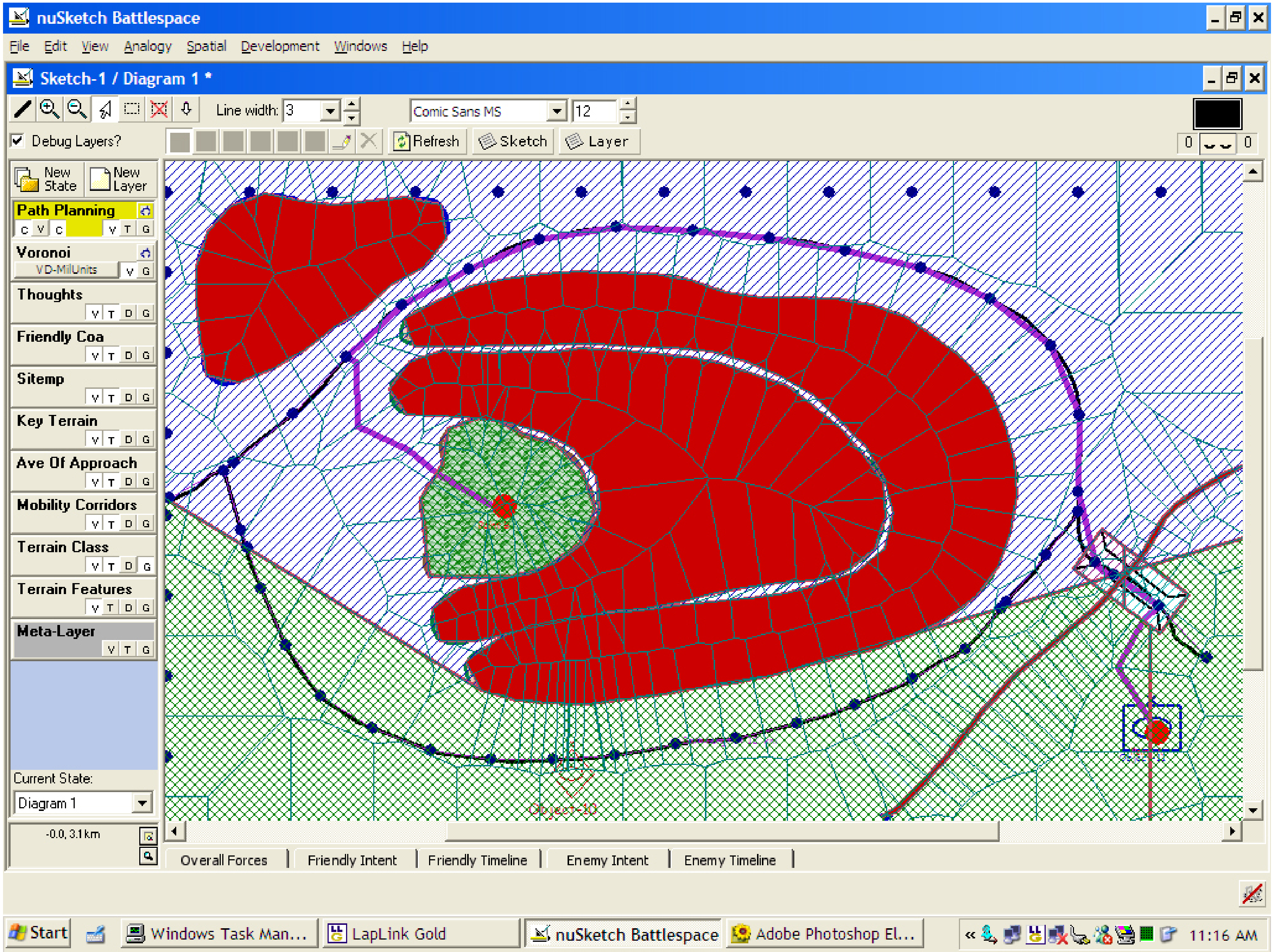
Free space represented as Voronoi



Visibility constraint regions



Intersect to compute constraint cells

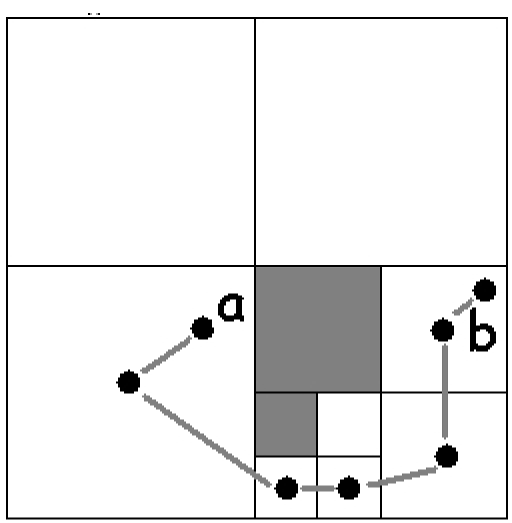


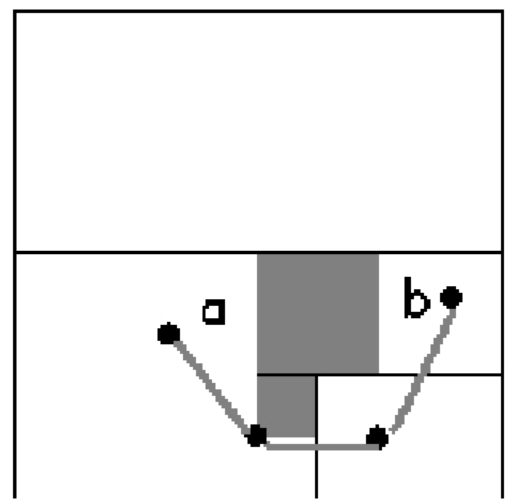
Modification: Iterative deepening

- Observation: For a search of depth *k*, there are many more nodes at the depth of *k* than the entire search tree for *k*-1
- Technique: Add a maximum depth of search
  - Start with small but semi-reasonable estimate
  - If failure, search again with larger maximum depth
- Tradeoffs
  - Search the same space near the start over and over again
  - Memory requirements can be dramatically smaller

Modification: Cleaning up paths

- With large pieces of space, paths generated can be unnatural
- Solution: Use a post-processing step to clean them up

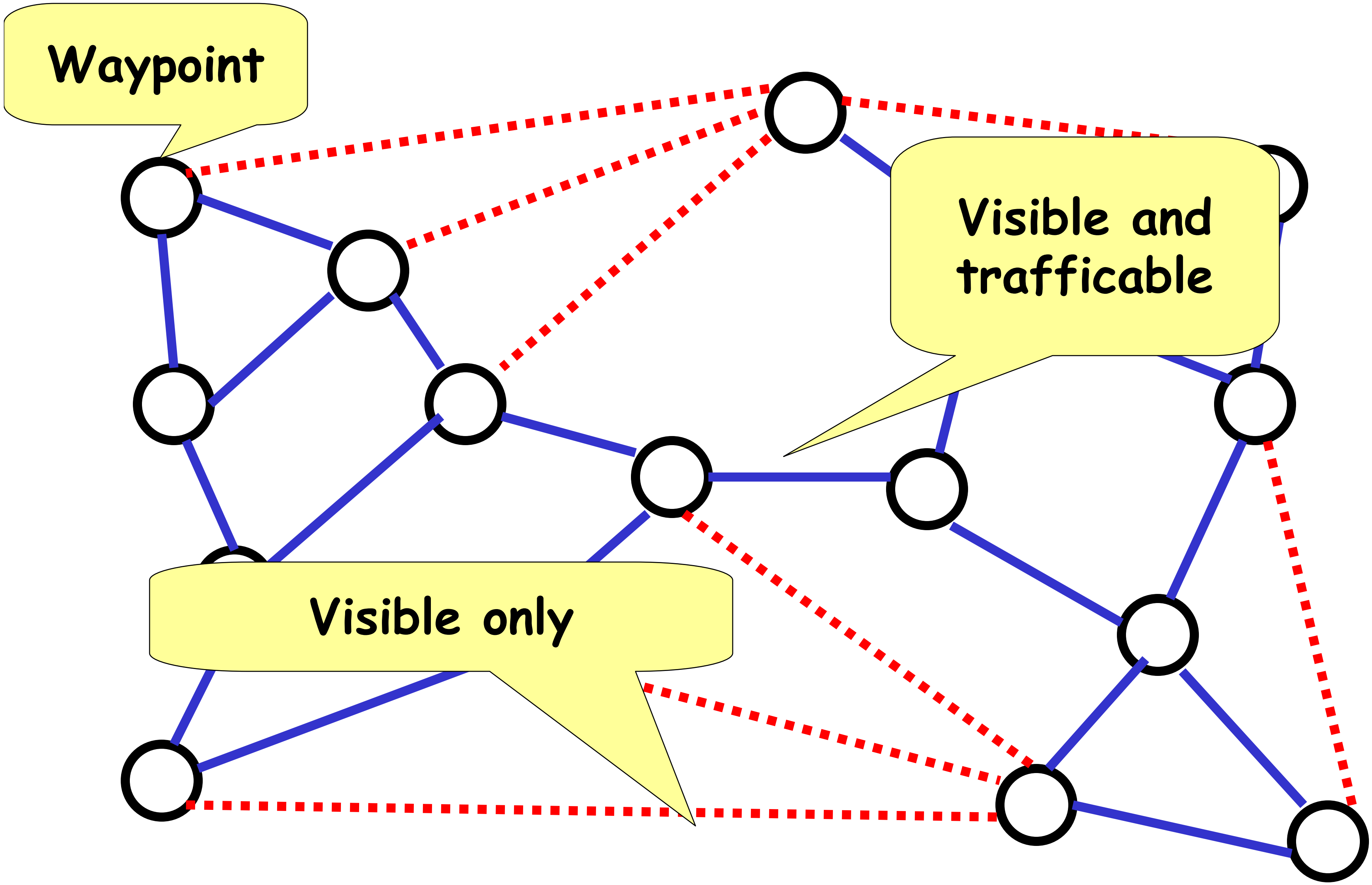




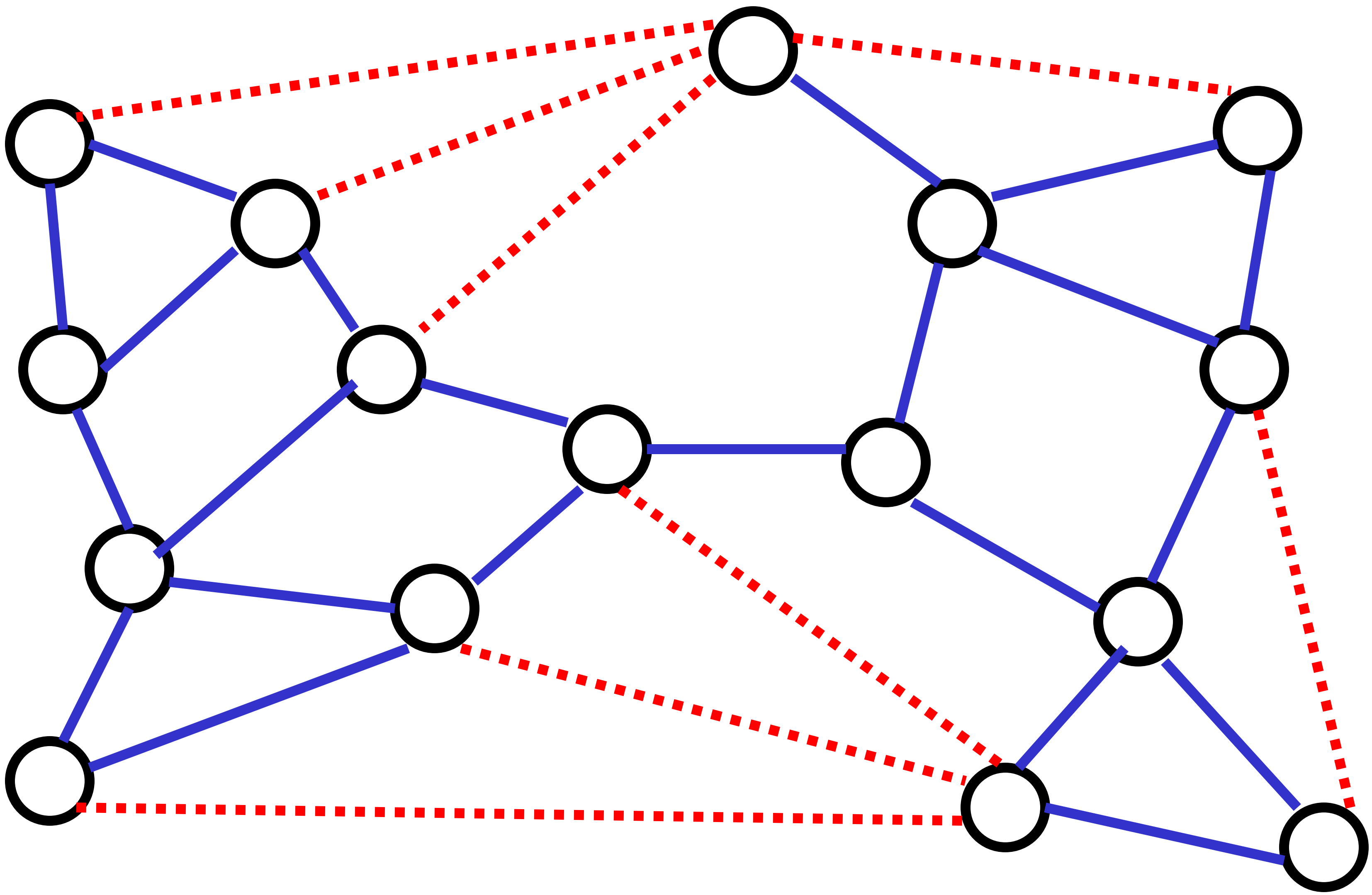
Position-Finding

- Many problems in games involving finding places
  - Where should I build cities?
  - How can I cut crime in my city?
  - Where could I ambush them?
  - Where would be a good place to hide?
  - What would be the most profitable place to put a ride?
- Position-finding = finding locations that satisfy (or optimize) particular criteria
  - Multiple constraints can be involved
  - What algorithms used depend on the available representations

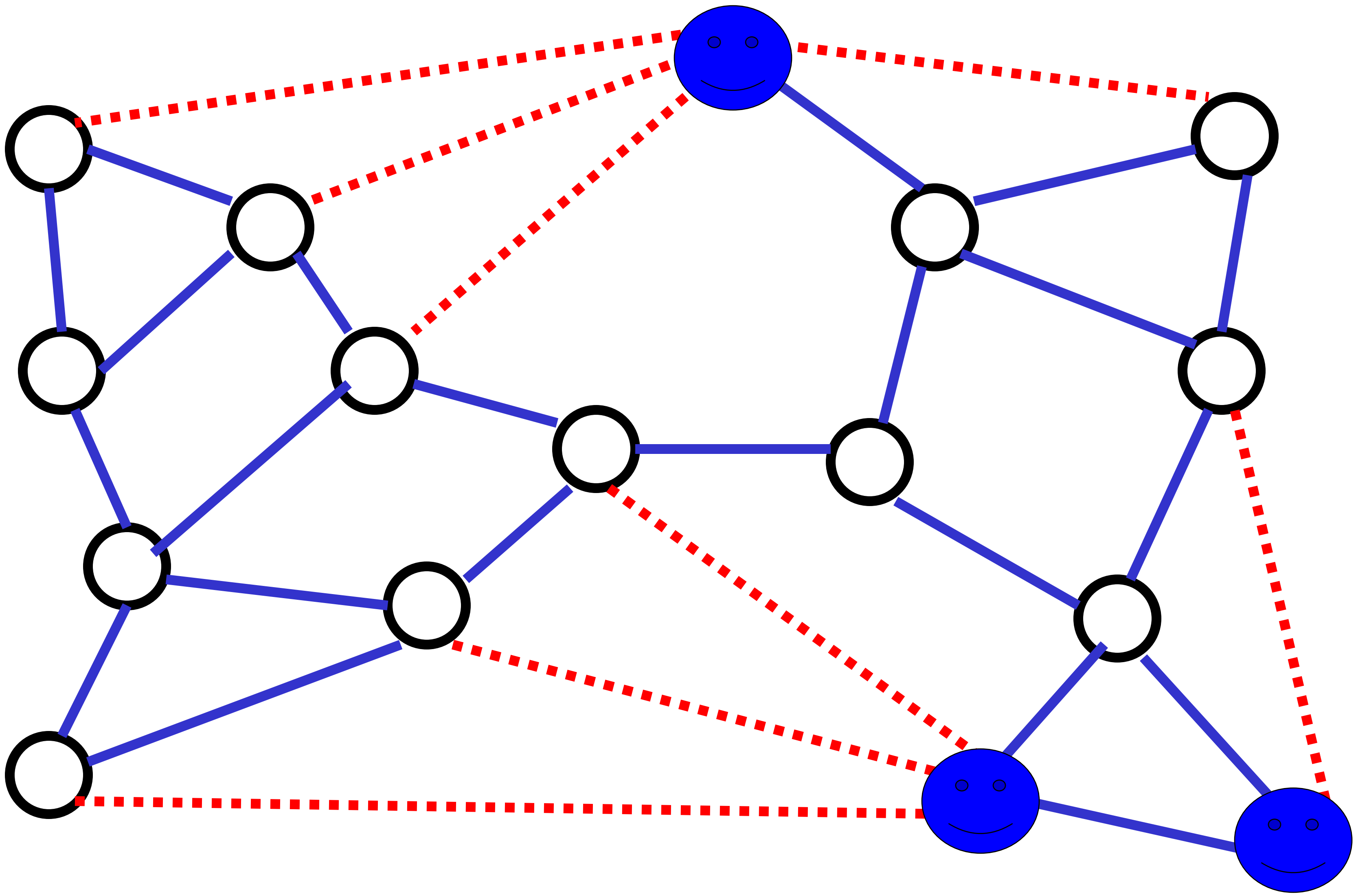
Example: Using waypoints for finding tactical positions



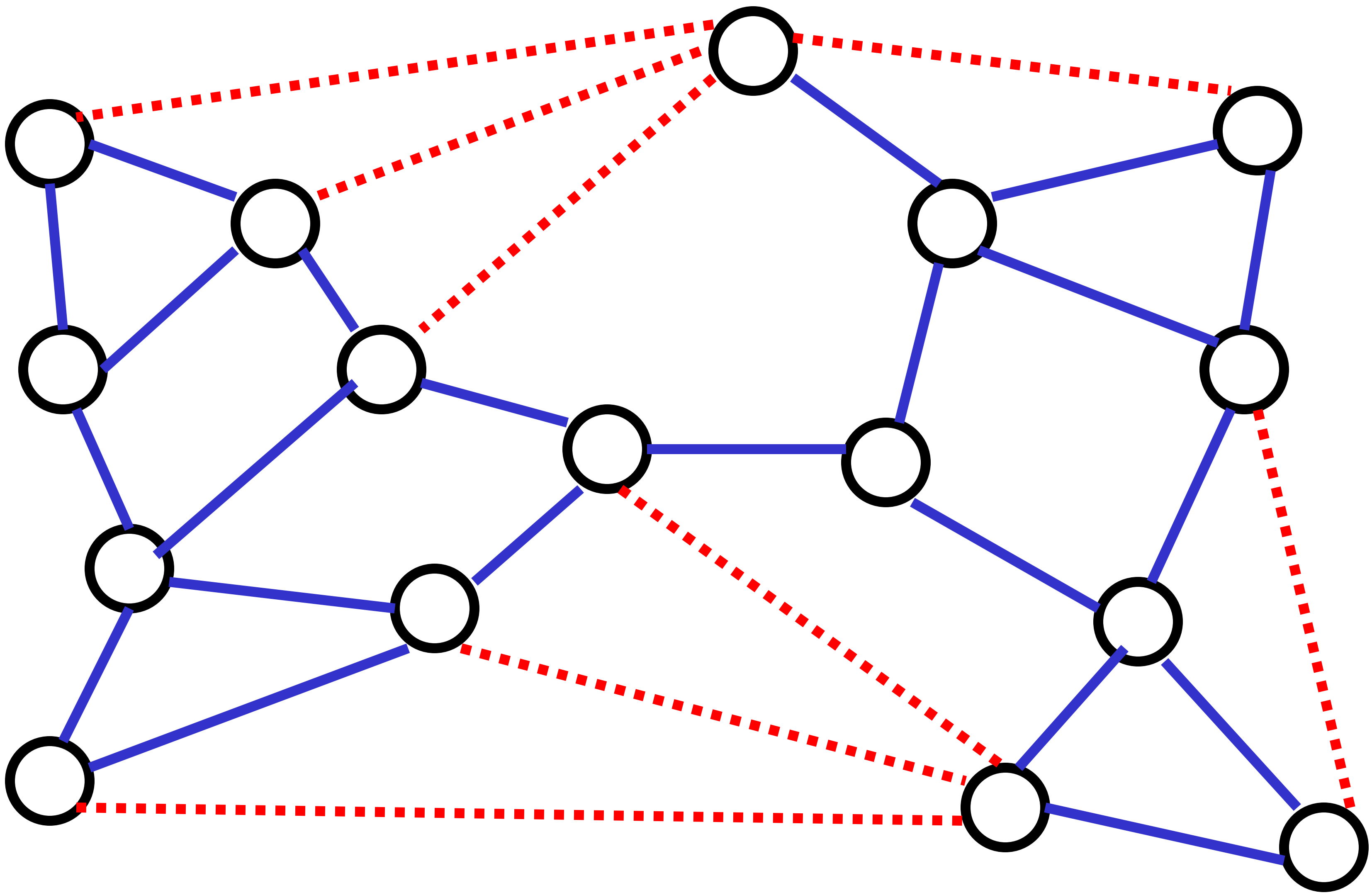
Where would be good sniper posts?



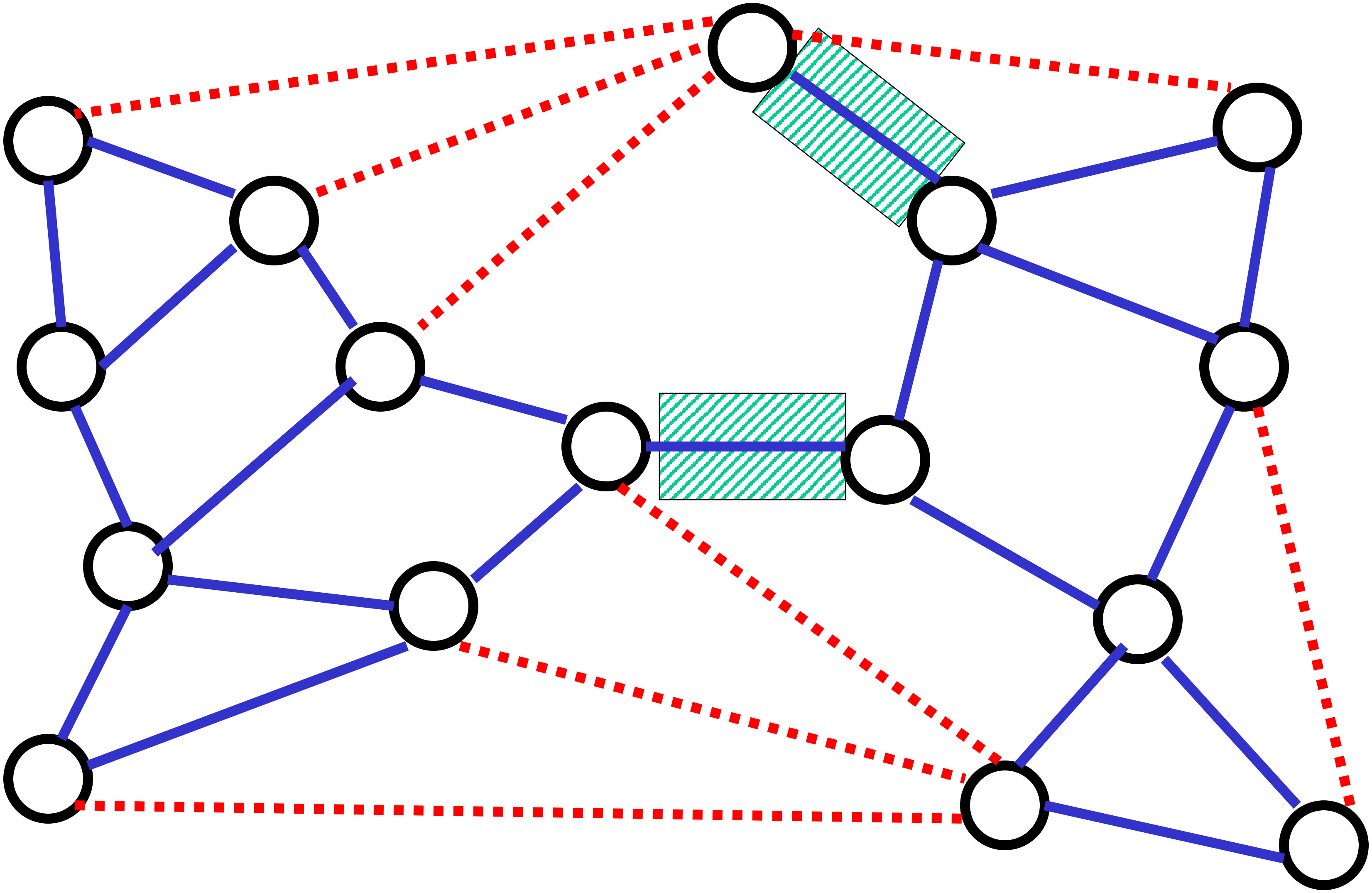
Where would be good sniper posts?



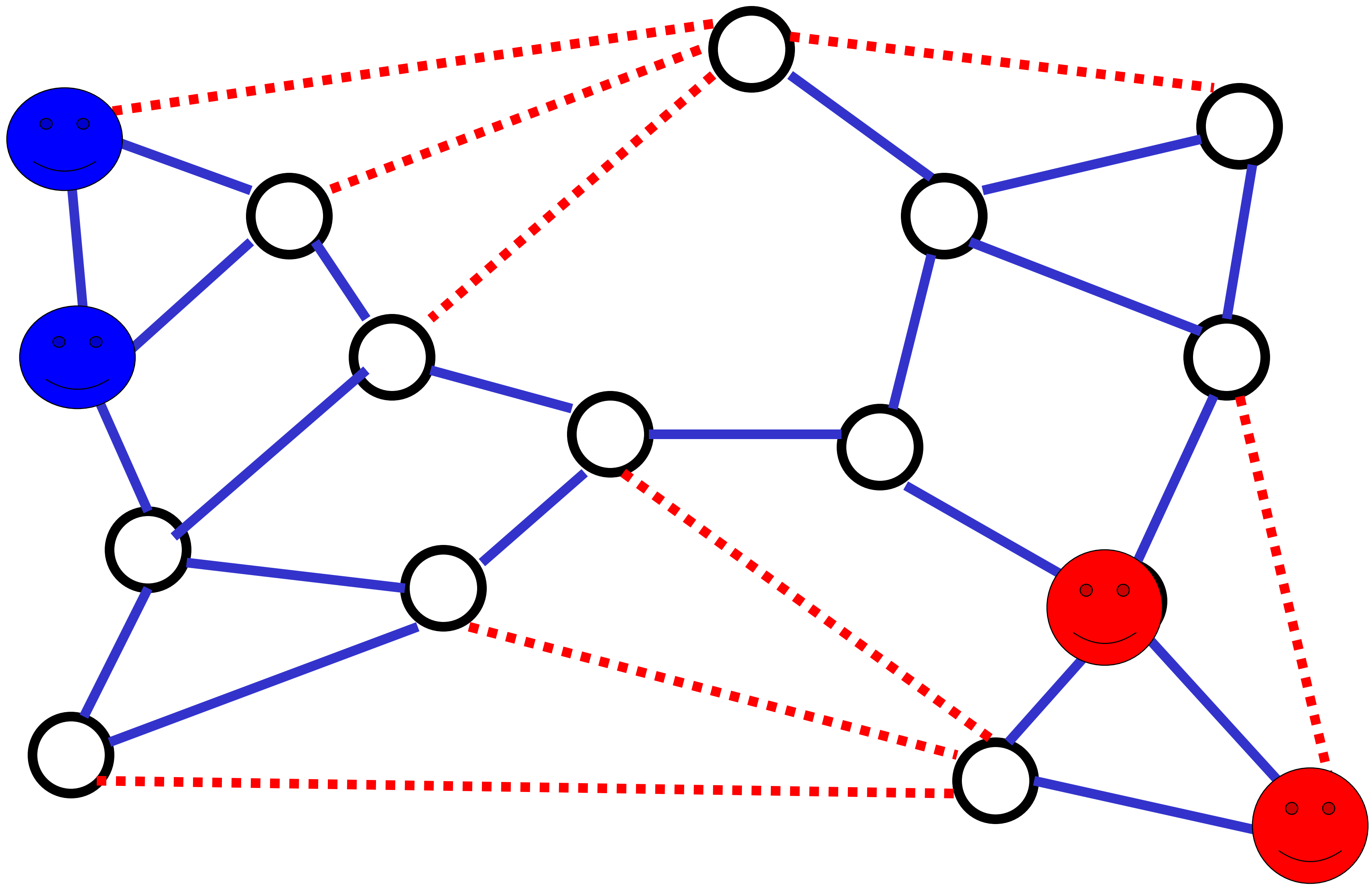
Where are the choke points?



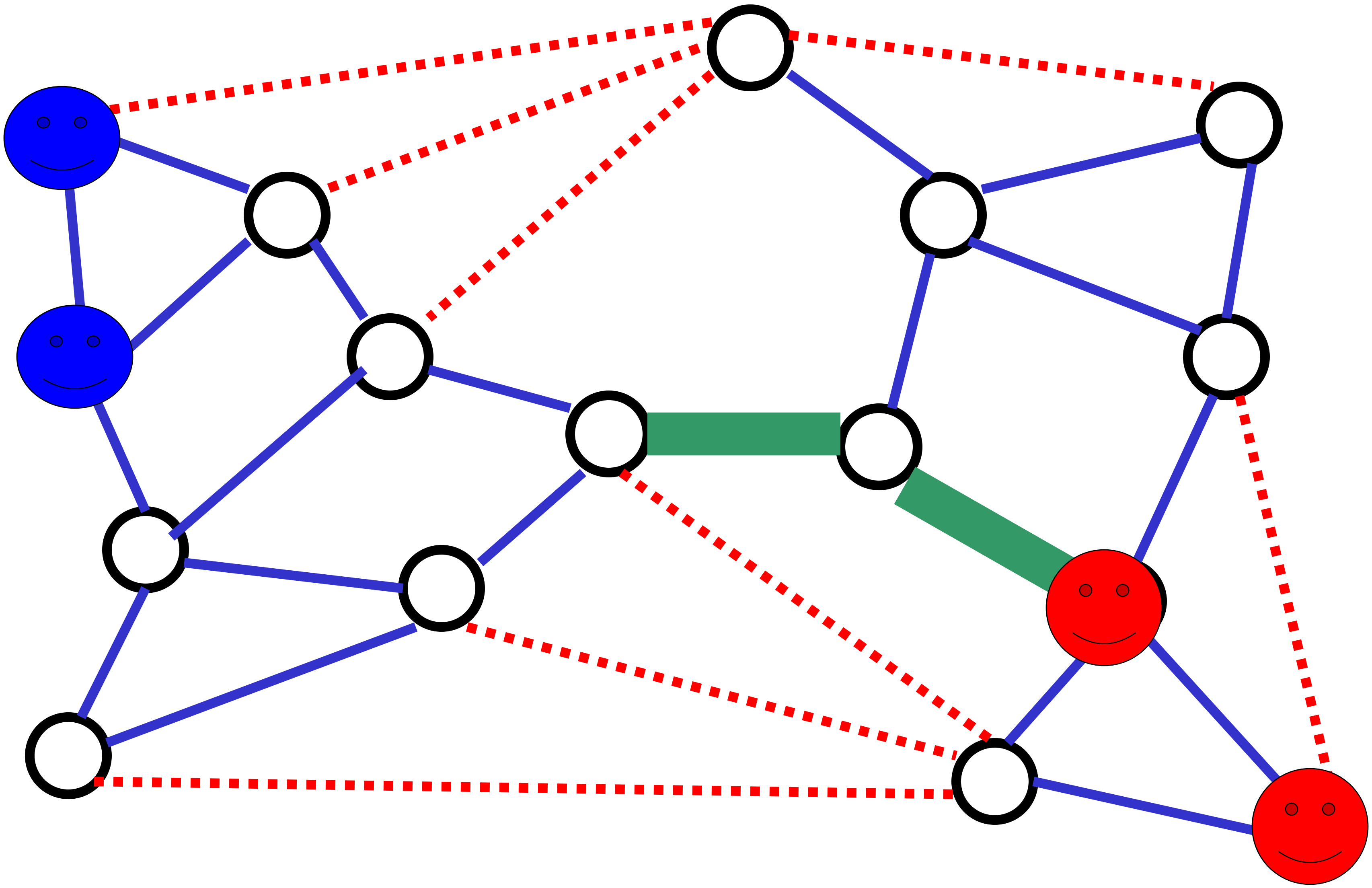
Where are the choke points?



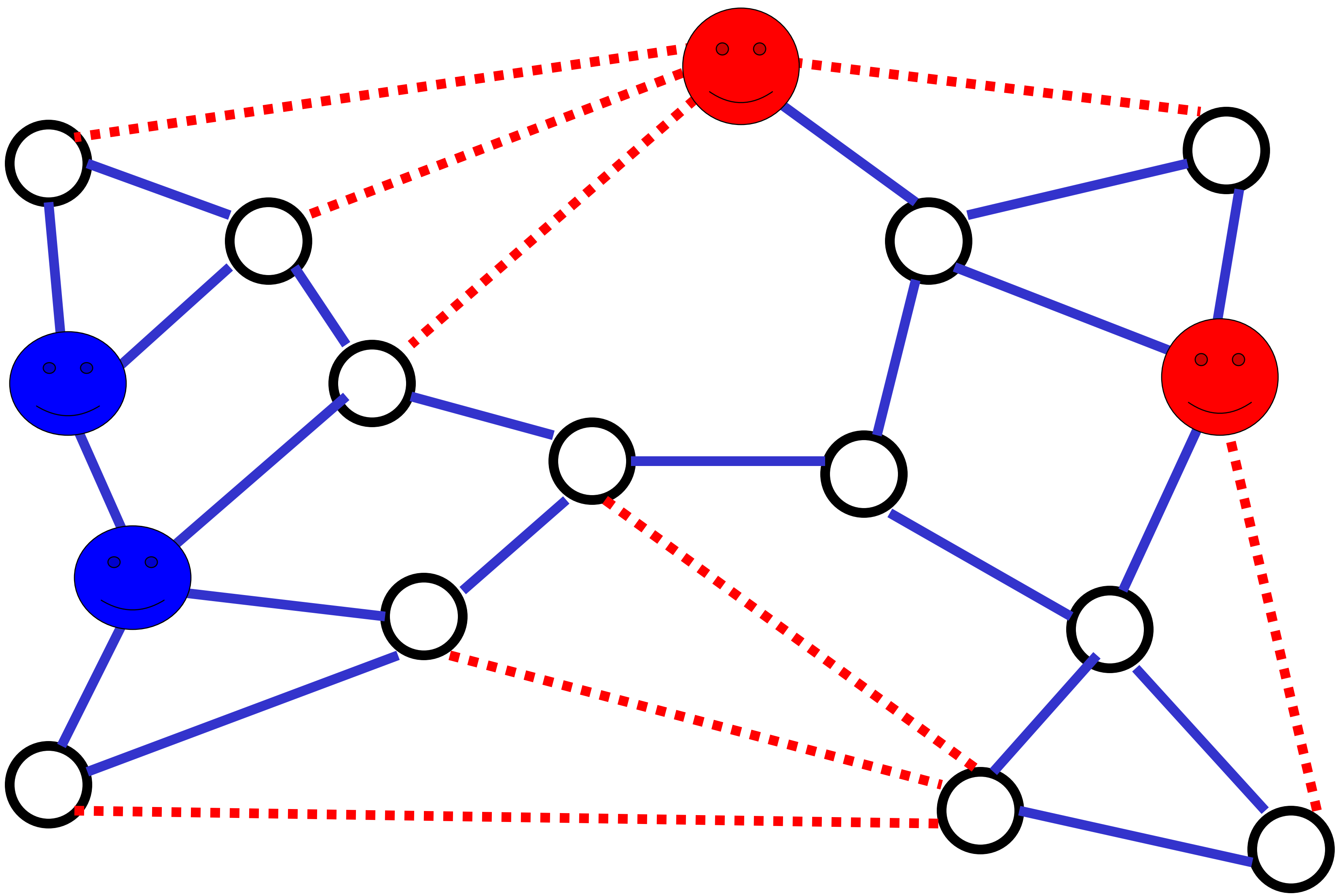
How would you sneak up on them?



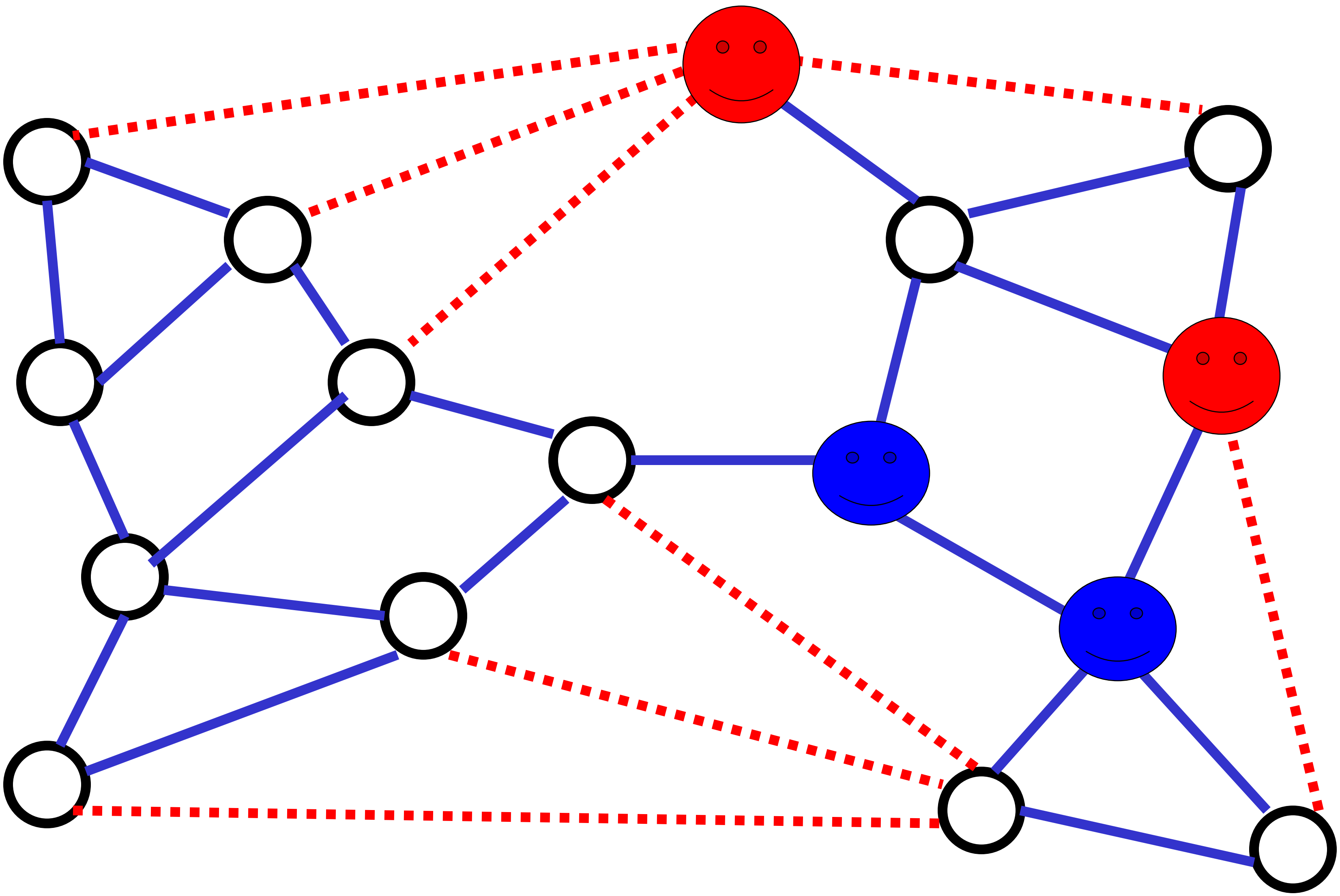
How would you sneak up on them?



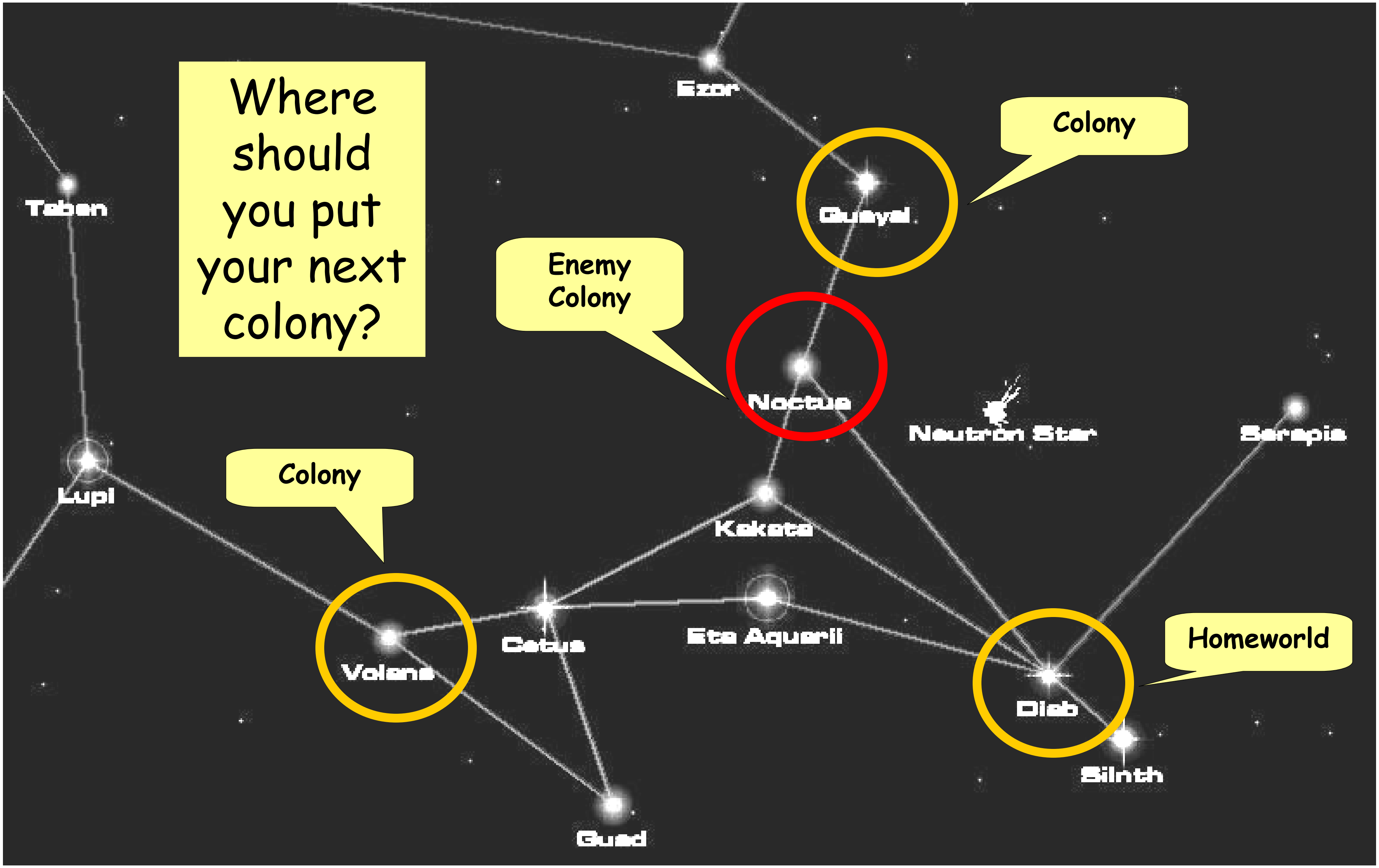
How would you trap them?



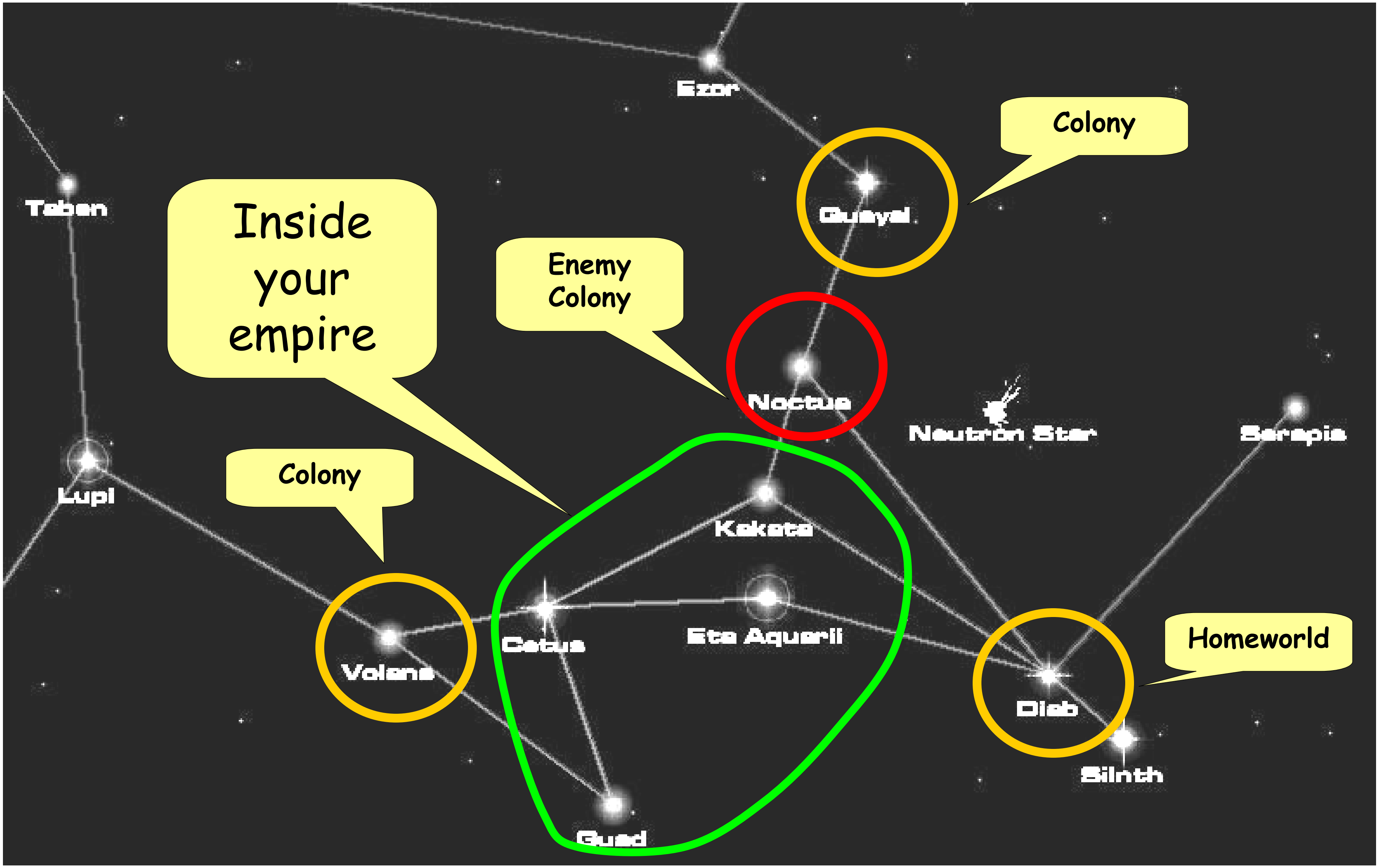
How would you trap them?



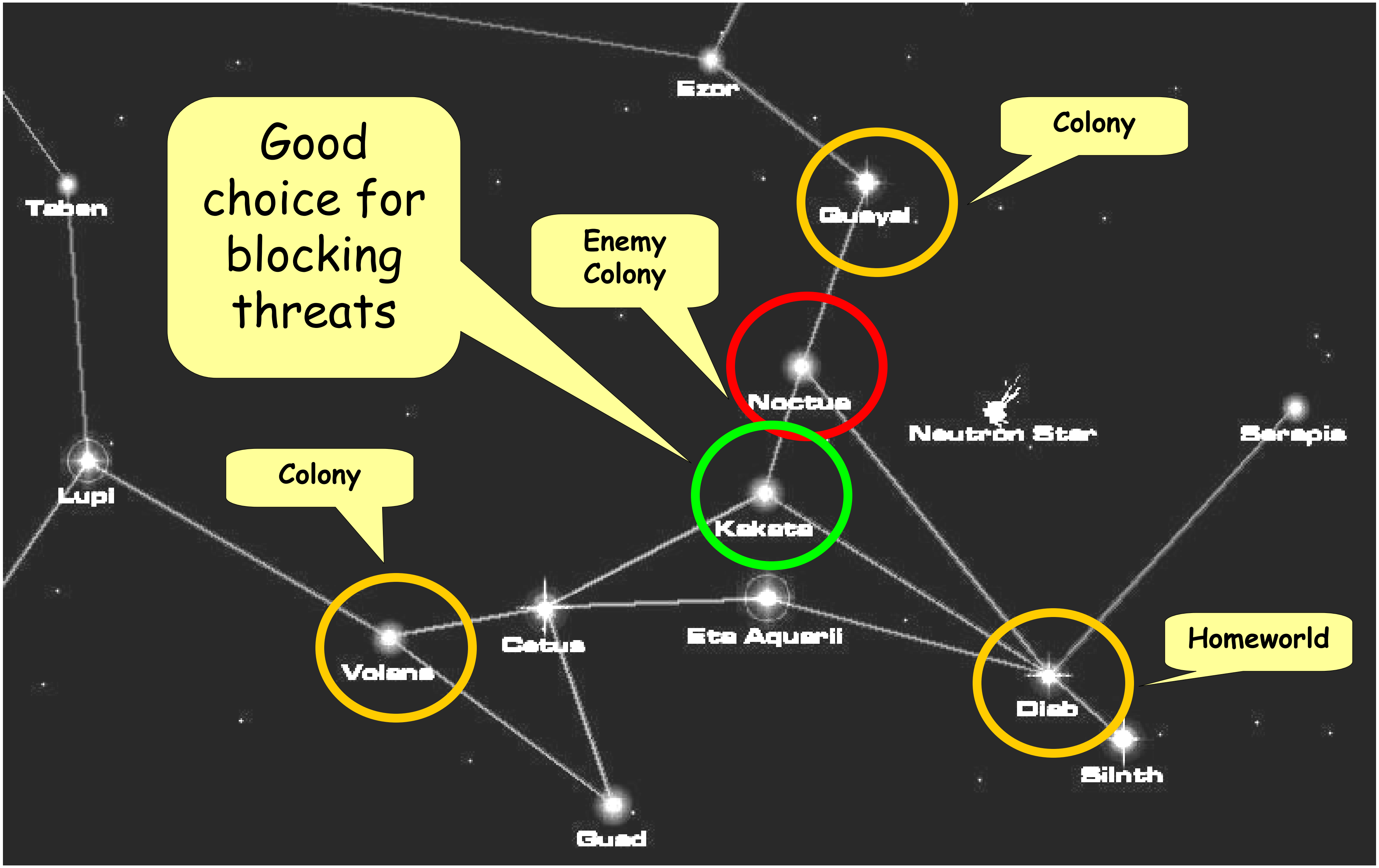
Example from MOO3



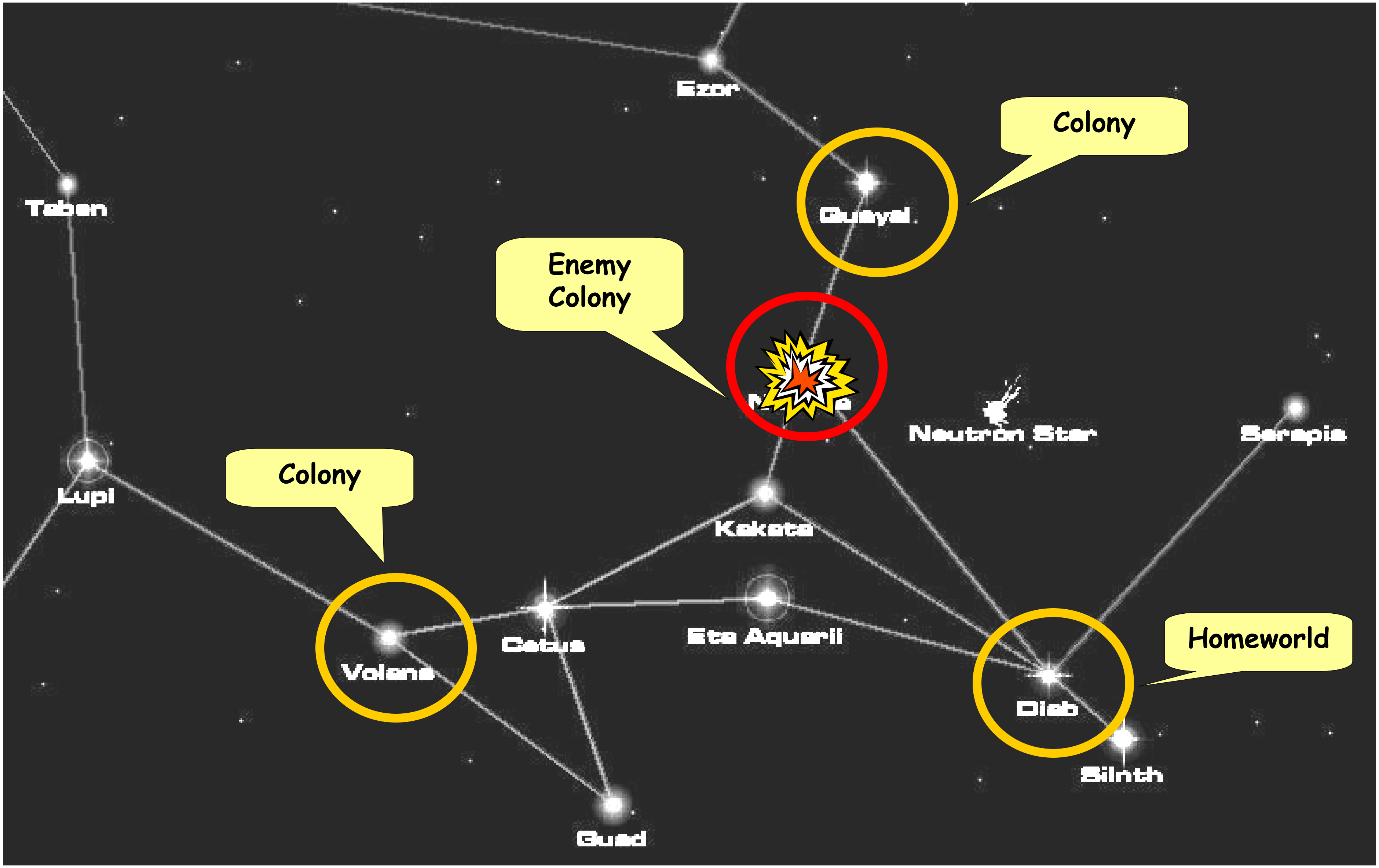
Understanding your terrain is key



Understanding borders prioritizes threats



Multiple ways to deal with threats



Influence maps

- Helps identify interesting positions on a map
  - Start from things of interest
    - E.g., friendly/enemy units, resources
  - Propagate numerical values to neighbors
    - Add/subtract values from different sources as appropriate
  - Analyze patterns of numerical values to select positions, boundaries
    - Where to put a mine or storage shed
    - The front between two warring nations

Demo

- http://www.ccg.leeds.ac.uk/james/influence/

Terrain Analysis Problems in FreeCiv

- What are they?
- How shall we solve them?

Exploration

- Solution 1: Cheat, AIs have full map knowledge
- Solution 2: Use a strategy for exploring the world

Creating the physical empire

- City site selection
- Infrastructure networks
  - Roads
  - Trade routes

War

- Where to attack your enemies
- How to use terrain to better defend yourself