FLINCH (in more detail)

MicroRTS implementation

Walter Gress

Consists of a decision/behavior tree and an A\* tree over that.

A\* Search: f(n) = g(n) + h(n)

g(n) is the cost of the path from the start node to n

h(n) is a heuristic function that estimates the cheapest path from n to the goal

s(n) fuzzy logic

k(n) the back-propagating reward/cost function

1. Zero out the graph and middle put all the deciders for each character
2. Take a random path down the tree evaluating each decider and reach behavior
3. If reward, propagate that, if punishment, propagate that.

At each decider the cost is calculated.And a sub-behavior is a behavior that is related to the leaf node. The branch to be chosen from the decider is that of the least cost. The overall cost for each decider is [0,1]. Deciders are decided first by measuring the cost from n to all of the leaf nodes. Then that is averaged over the number of leaf nodes.

The next is the heuristic function.. Intiiallly the heuristic function tries every possible set in the tree, like the full space search. It can stop the search at any time. If the current possiblity is already better than the search already found. h(n) returns the least path down the tree.

k(n) is backpropagating. At the leaf node the behavior is rewarded or punished by giving a high cost and rewarding by giving a low-cost.

Finally stats. Each decider compares against a static set of variables and for example:

1, Reset soldier

2. Go down random path

3. Path consists of neutral deciders,

4. Back propagate k(n), Hard-coded to reward (reduce cost on the back path or hard-code to punish (increase cost on the back path)

5. At root.

6. Decider. Calculate the decision. Calculate g(n), h(n). In this case

g(n) = 0.03

h()n) = 0.02

s(n) = 0.04

k(n) = -0.5

Total cost: 0.4.

The propagation function is what keeps the search from blowing up. That way the cost isn’t always 1.0 (this was a problem before).