Forager Allocation in Bees and Parallel Double Depth-First Search

Lecture 36
CS 312

Objectives

• See that designing new algorithms is fun and interesting.
• Emergent systems are systems in which global behavior emerges from the behavior of individuals.

Outline

• How bees assign foragers to nectar patches.
• Non-emptiness test for Buchi automata using double depth first search.
• How to do double depth first search in parallel using bees as a model.

Forager Allocation Problem

• Given:
  – several foraging sites with varying quality
  – the current needs of the hive
  – a pool of foragers
• Allocate the foragers to:
  – maximize the collection of needed resources
• Without central control.
Scouts multicast their findings

Scout performs a waggle dance which is observed by unemployed workers. Duration and intensity of the dance describes the quality of the dance.

Workers visit patches.

A worker will observe a dance and leave for a patch.

Forager Allocation

- No central control
- Quickly adapts to change
- No other distribution accumulates value more than twice as well
- Each bee accumulates value at the same rate as they would with global knowledge.

Double Depth First Search

- More efficient version of nested depth first search.
- Used to check emptiness for a Buchi automata

Emptiness as Correctness

\[ L(m) \cap L(s) = \emptyset \text{ if the actual behaviors are correct} \]

Verification Steps

1. Build a Buchi automata representing the system under test
2. Build a Buchi automata representing the opposite of the allowed behaviors
3. Use double dfs to see if the intersection is empty.
4. If empty then correct, else incorrect.
Buchi Automata

Just like a DFA, except:
- Every state has an outgoing edge
- Accepts infinitely long words
- Accepts a word if it passes through the accept state infinitely often.

Double Depth First Search

Perform a depth first search for an accept state

When an accept state is found, launch another search to find a loop which includes the accept state. The loop allows the accept state to be visited infinitely often.

Double DFS in Parallel

- In all double DFS implementations, a hashtable is used to store visited states.
- The hashtable is split across nodes in a network.
- This requires coordination of efforts to find a cycle.

Decentralized Double DFS?

- Can something similar to forager allocation in honey bees be used to develop a double DFS algorithm without a global hashtable?
- Bees do quite well without global knowledge. Can double DFS?
Double DFS in Parallel

First, send out scout processes to find bad states.

Double DFS in Parallel

First, send out scout processes to find accept states.

Double DFS in Parallel

Scout multicasts accept state to unemployed foragers.

Double DFS in Parallel

Foragers look for a path back to accept state.

Double DFS in Parallel

This Buchi automata represents a non-empty language.

The Next Step

- Implement the algorithm.
- Determine if the scout/forager organization finds cycles faster than parallel processes searching at random.
- Obtain fame, glory and tenure.
- Interested? send mail to jones@cs