COMP 250

Lecture 34

graph applications

- garbage collection (mark and sweep)
  - Google search (pagerank)

Nov. 30, 2016
Garbage Collection

Dog  myDog = new Beagle(“Bob”);

myDog = new Terrier(“Tim”);

Nothing references the Bob the Beagle.

Bob is wasting memory.  Bob is garbage.
Dog   myDog = new Beagle(“Bob”);

myDog = new Terrier(“Tim”);
Dog  myDog = new Beagle(“Bob”);

myDog = new Terrier(“Tim”);

Bob is garbage.
As the program continues, more “garbage” accumulates.
Let’s ignore the call descriptors from on today.
Every object has a location in memory: in Java, the location is a 24 bit number (address).

Call Stack

- main()
- mA()
- mB()
- mC()
The Java Virtual Machine (JVM) also maintains a linked list of all objects.
Q: What to do when object space fills up?

A: Let the program crash.

A: Reuse the space we don’t need. (Garbage collection)
“Live objects” (not garbage): referenced either from a call stack variable or from an instance variable in a live object.
Q: If these objects are only referenced by each other, then are they garbage?
A: Yes, because they will never be used by the program.
Garbage collection: “Mark and Sweep”

1) Build a graph, and identify live objects (“Mark”)

2) Remove garbage (“sweep”)
Build a graph that corresponds to this:

Call Stack

Vertices correspond to reference variables in call stack, and to objects.

Edges correspond to are references.
Traverse the graph, starting from each vertex that corresponds to a reference variable on the call stack.

Visiting a node = *mark* it as live.
Phase 1: “Mark” the garbage

Call Stack

main()

mA() → mB() → mC()

Terrier object “Tim”

Beagle object “Bob”

another garbage object
Phase 2: “Sweep” the garbage

Call Stack

mA()
mB()
mC()
main()
Using a complementary list that keeps track of **free space**. JVM uses the free space to make new objects.
After garbage collection, continue execution

• New objects can be added, where there is a big enough gap in free space.

• Garbage collection needed again when there is no gap big enough.

• Program needs to stop (temporarily) to do garbage collection. This is not good for important real time applications.
COMP 250

Lecture 34

graph applications

- garbage collection (mark and sweep)
  - Google search (pagerank)

Nov. 30, 2016
Google page rank

Q: How important (high rank) is a web page?

A: Loose recursive definition of importance:
   - a web page is important if other important web pages link to it.
Q: How important is a web page $v$?

A:

- Which set of pages $\{w\}$ link to $v$ and how important are these pages?

- How many other pages does each $w$ point to?
Let \( N_{out}(w) \) be the ‘out degree’ of \( w \).

Define \( R(v) \) to be the pagerank of \( v \).

\[
R(v) = \sum_{(w,v) \text{ is Edge to } v} \frac{R(w)}{N_{out}(w)}
\]

This is recursive, but there is no base case. Solve it using other linear algebra methods (uniform initial guess and then iterate).
What does Google Search Engine do?

- Download all reachable pages on web
- Build (update) the graph
- Compute page rank

What do you the user do?

- Enter query words and ask the search engine for pages containing these words

What does Google Search Engine then do?

- Delivers a list of pages ordered by page rank
What does Google Search Engine do?

• Web crawler downloads all reachable web pages
• Build a graph (update)
• Compute page rank for each web page
What does Google Search Engine do?

- Web crawler downloads all reachable web pages
- Build a graph (update)
- Compute page rank for each web page
- Build a map:
  - Values: list of web pages (URLs) containing keyword
What do you the user do?

• Enter query words (keywords)

“Montreal”
“bicycle”
What does the search engine do?

- Get the value for **each keyword**.
- Compute the intersection of these lists of web pages and list in order of pagerank.

**Values:** list of web pages containing each key

**Keys**

- "Montreal"
- "bicycle"
For more details, including the original research papers describing page rank, see Sergey Brin’s old home page from the late 1990’s:

http://infolab.stanford.edu/~sergey

Google’s other founder is Larry Page (no pun intended)

https://engineering.stanford.edu/about/heroes/larry-page

Sergey Brin's Home Page

Ph.D. student in Computer Science at Stanford - sergey@cs.stanford.edu

Research

Currently I am at Google.

In fall ’98 I taught CS 349.

Data Mining