Announcements

Outline

- *Intelligent file hoarding for mobile computers* Carl Tait, Hui Lei, Swarup Acharya and Henry Chang

  Mobicom '95

Hoarding

- Prefetching and storing objects for access when the network is disconnected or weakly-connected

- E.g. Internet Explorer lets you browse web pages Offline. Avantgo lets you access web pages offline in PDAs. I download power point slides for this presentation into the laptop

Problem

- Local storage space is limited. How do I hoard all the data that I need so that I can continue to work, while I'm offline?
  - Suppose I have 80 GB worth of files in the connected machine. I have a 20 GB hard drive in the mobile computer, how can I copy files such that I can continue to work without disruption?
  - Suppose I have 80 GB hard drive in the mobile device, hoarding profile is not necessary.

Problem

- User input is necessary.
  - Users should say what they want to be hoarded for disconnected access. But, users do not know what files they really need. They can tell that they need Powerpoint. But they can’t say that they also need some .dll file that is needed to print the presentation onto a laser printer (say)

Approaches to create hoard profiles

- Do Nothing:
  - Run the application that you intend to use on the road. Normal caching mechanism would've stored the documents for later access
  - Problem is that, there are no "typical" runs. Every program run is different, just because it worked while connected does not mean that it will work while disconnected

- User Provided Information
  - Coda uses this approach. The users specify the files that they want to be hoarded.
  - Most of the time, users do not even know what files they need. Problem of hidden files and critical files that are rarely used

- Snap-shot spying:
  - System keeps track of all files access between some markers. You specify <START SPYING> and <END SPYING> messages to refine files.

- Semantic Distance:
  - Seer system looked at the semantic distance between files to predict related files

Transparent, analytical spying

- Automatically detect working sets for applications and data
- Provide generalized bookends to delimit periods of activity
- Present convenient bundles to the user using a GUI
- Let the users “load the briefcase”

Detection of working set

- Working set defines that set of files that are part of a program execution. This may include data files, shared libraries (DLLs), related applications
- Log file accesses to create program execution tree. Subsequent executions of program help refine this tree
- Problem is differentiating application vs user data, you want to learn the “application” from my executions of the application, not the files that I operated on

Differentiate application and data

1. File name extensions (.exe, .bat, .ini etc)

2. Directory inferencing (/usr/include/file.h is related to application, /home/surendar/data/foo.dat is related to data)

3. Timestamps – Files modified at the same time as application probably belong with application

Generalized bookends

- Measure all files and applications accessed during the time intervals
- Generalize data over multiple executions
- Presentation and Selection – GUI tools

Implementation issues

- Data orphan problem
  - Local program accesses remote data
- Program orphan problem
  - Remote program accessing only local data
- GUI tools to select files for hoarding

Discussion

- As disks become cheaper, smaller and larger, is hoarding needed?