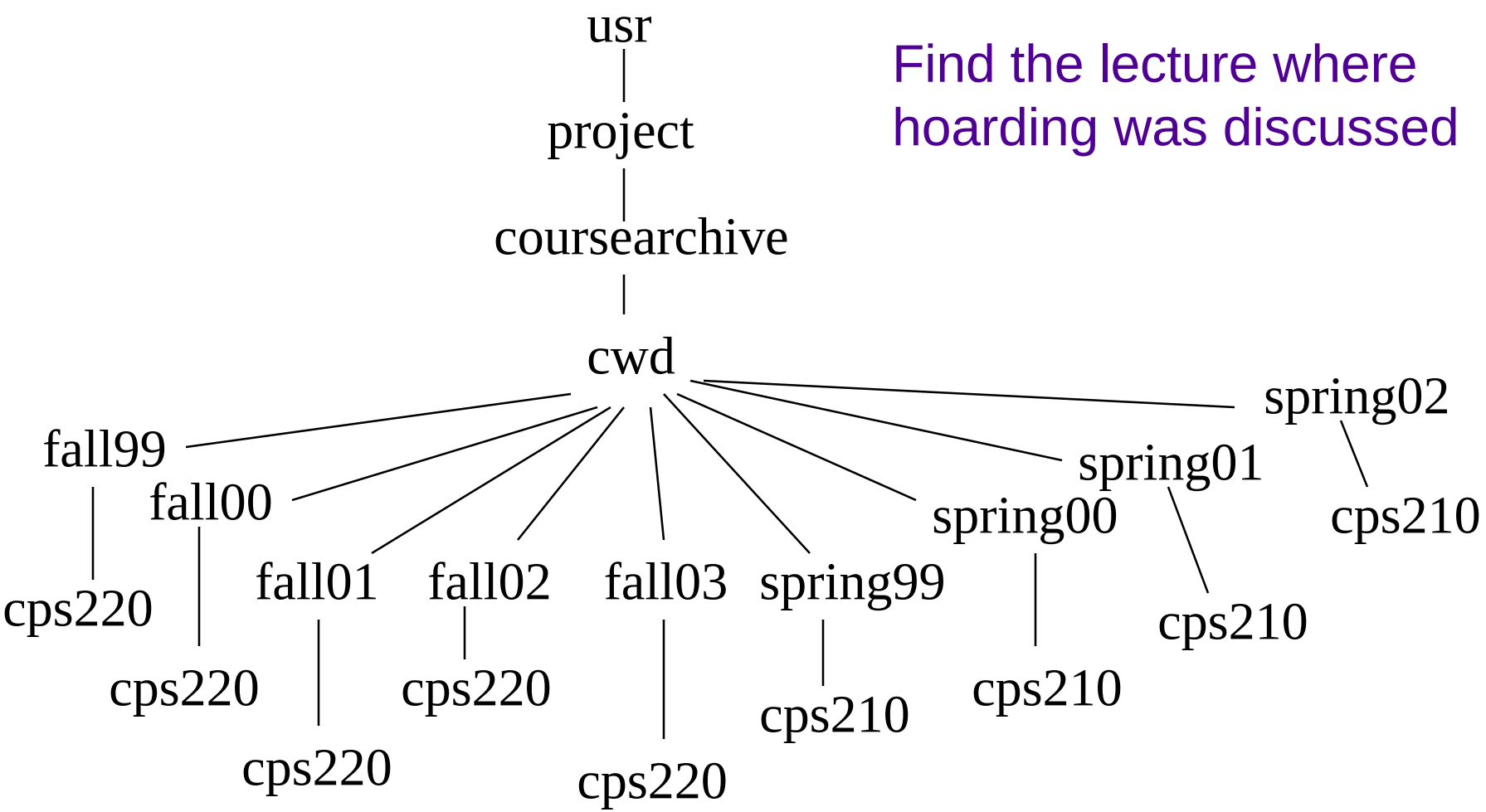
Outline

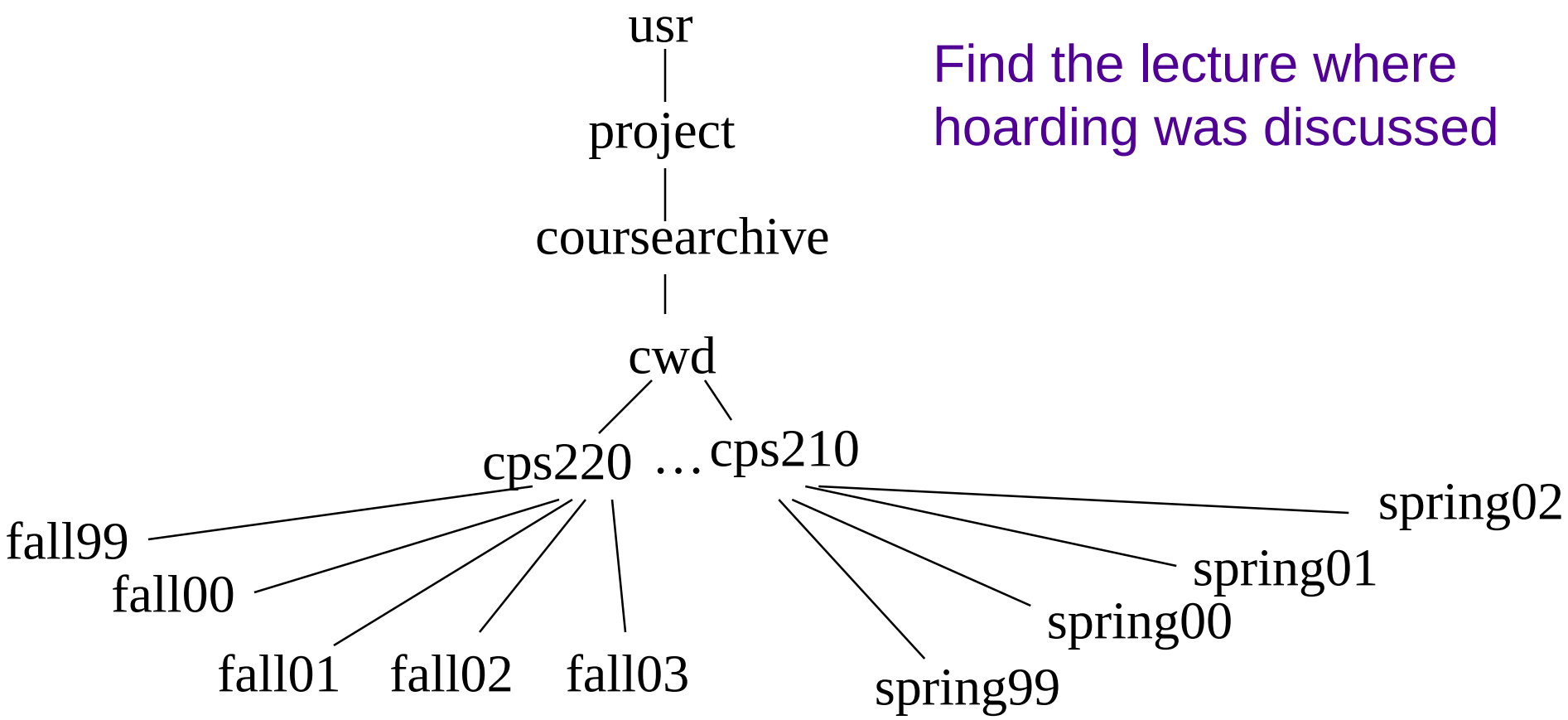
- Objective
  - Attribute file naming:
    “Why can’t I find my files?”
  - Hoarding techniques and their use in disconnected file systems
- Administrative
  - New programming project up
  - Revisions in schedule – I’m out of town on April 13

An Example of the Problem



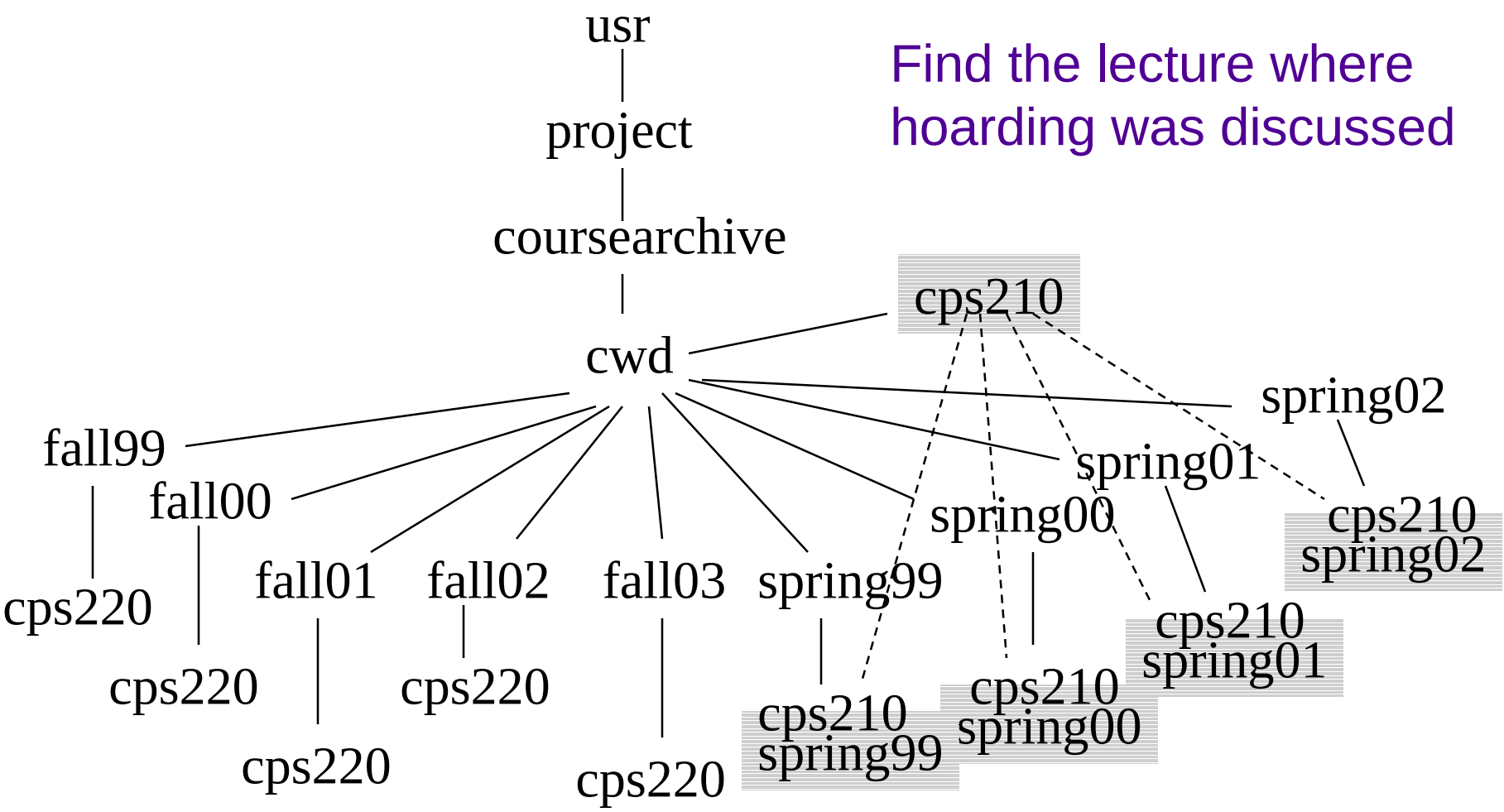
Find the lecture where hoarding was discussed

An Example of the Problem



Find the lecture where hoarding was discussed

An Example of the Problem



An Example of the Problem

- It gets worse:

  /home/home5/carla/talks

  2 laptops (one lives at work, one at home)

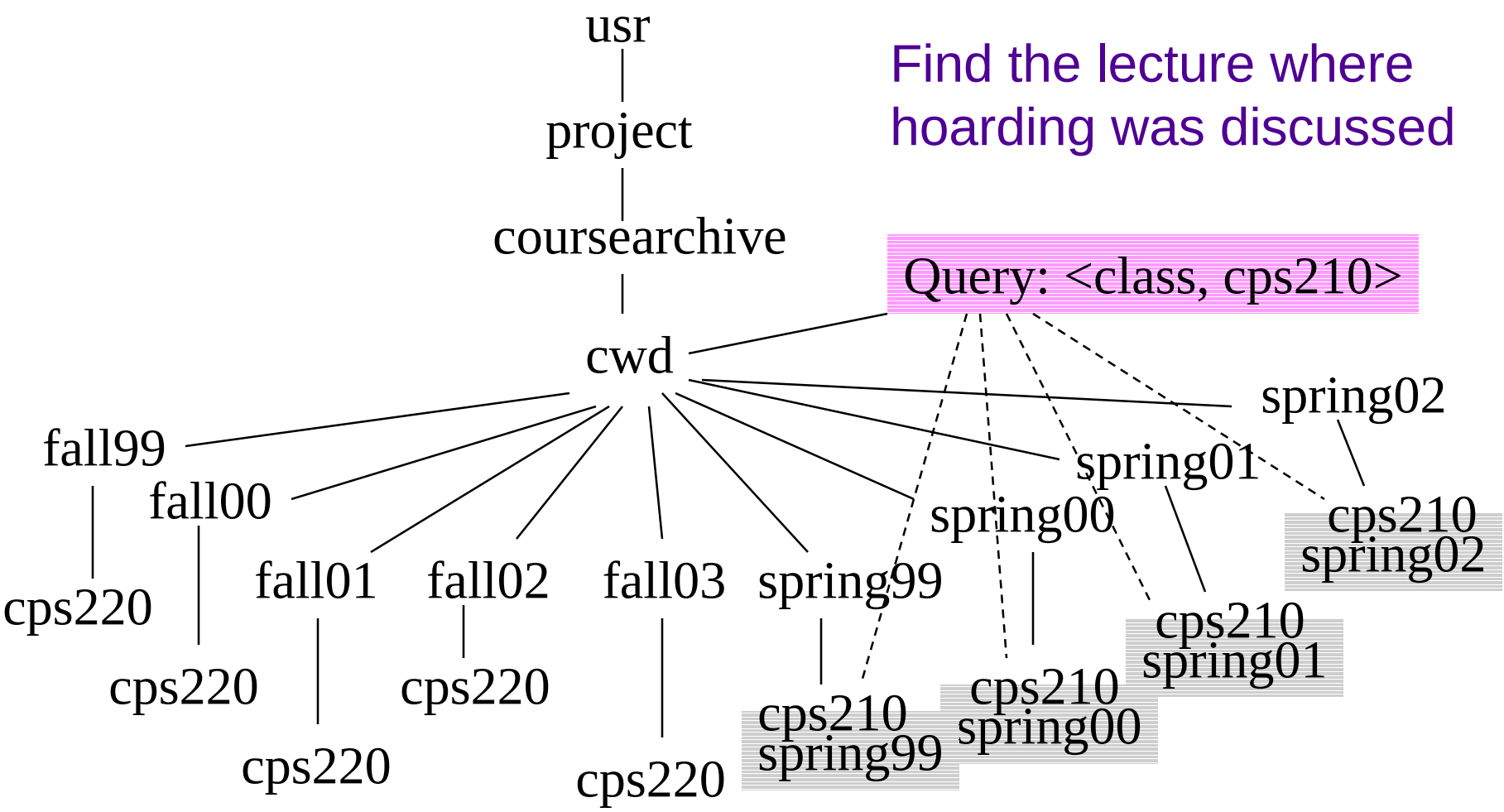
  desktop machine at home

- Forest not a tree!
  - Growing more like kudzu

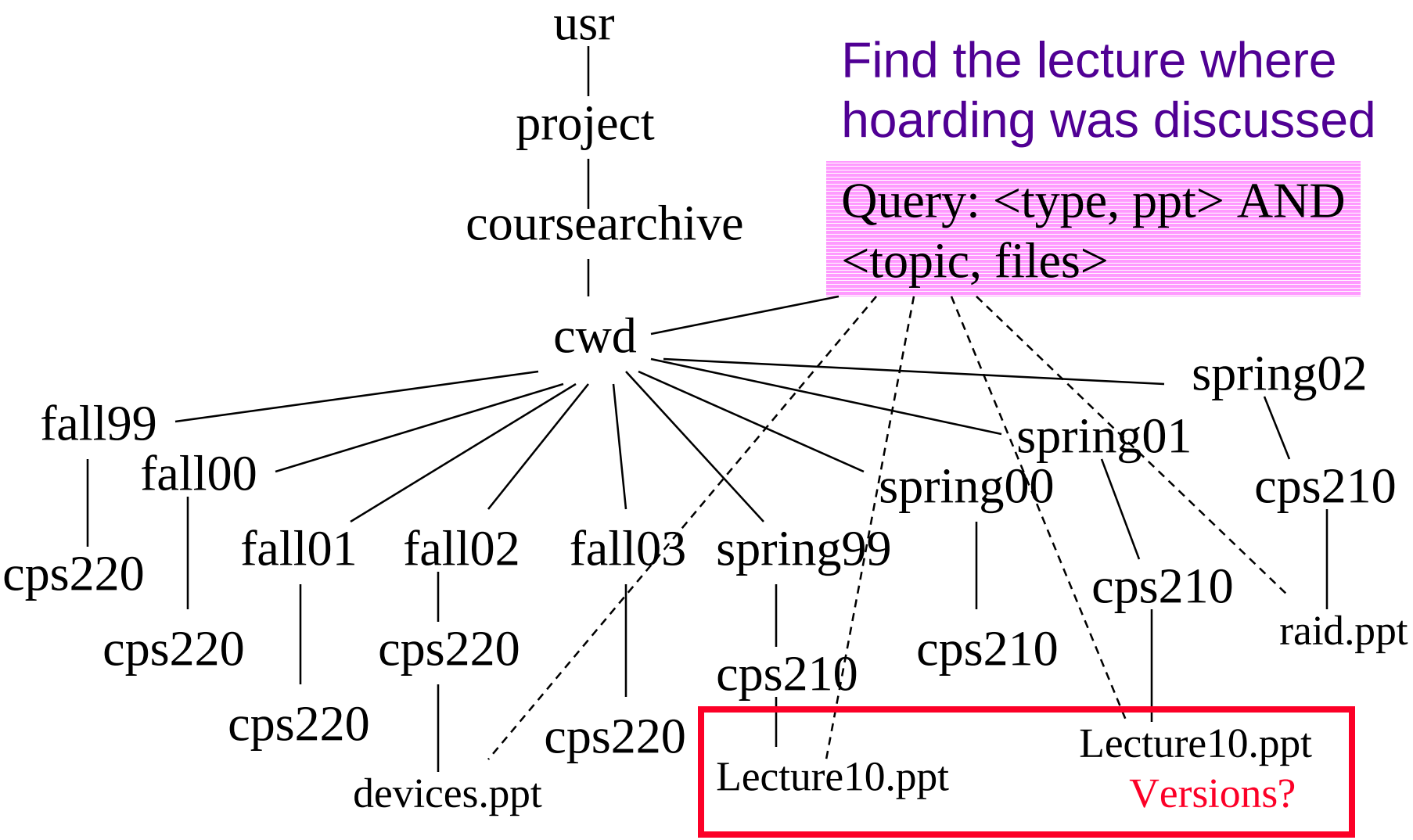
Attributes in File Systems

- Metadata: <category, value>
- How to assign?
  - User provided – too much work
  - Content analysis – restricted by formats
    - Semantic file system provided transducers
  - Context analysis
    - Access-based or inter-file relationships
- Once you have them
  - Virtual directories – “views”
  - Indexing

Virtual Directories



Virtual Directories



Issues with Virtual Directories

- What if I want to create a file under a virtual directory that doesn't have a path location already?
- How does the system maintain consistency? We should make sure that when a file changes, its contents are still consistent with the query.
  - What if somewhere a new file is created that should match the query and be included?
  - What if currently matching file is changed to not match?
- How do I construct a query that captures exactly the set of files I wish to group together?

Example: HAC File System
(Gopal & Manber, OSDI99)

- Semantic directories created within the hierarchy (given a pathname in the tree) by issuing a query over the *scope* inherited from parent
  - Physically exist as directory files containing symlinks
- Creates symbolic links to all files that satisfy query
- User can also explicitly add symbolic links to this semantic directory as well as remove ones returned by the query as posed.
  - Query is a starting point for organization.
- Reevaluate queries whenever something in scope changes.
- Uses Glimpse – an index-based file search tool for content-based queries.

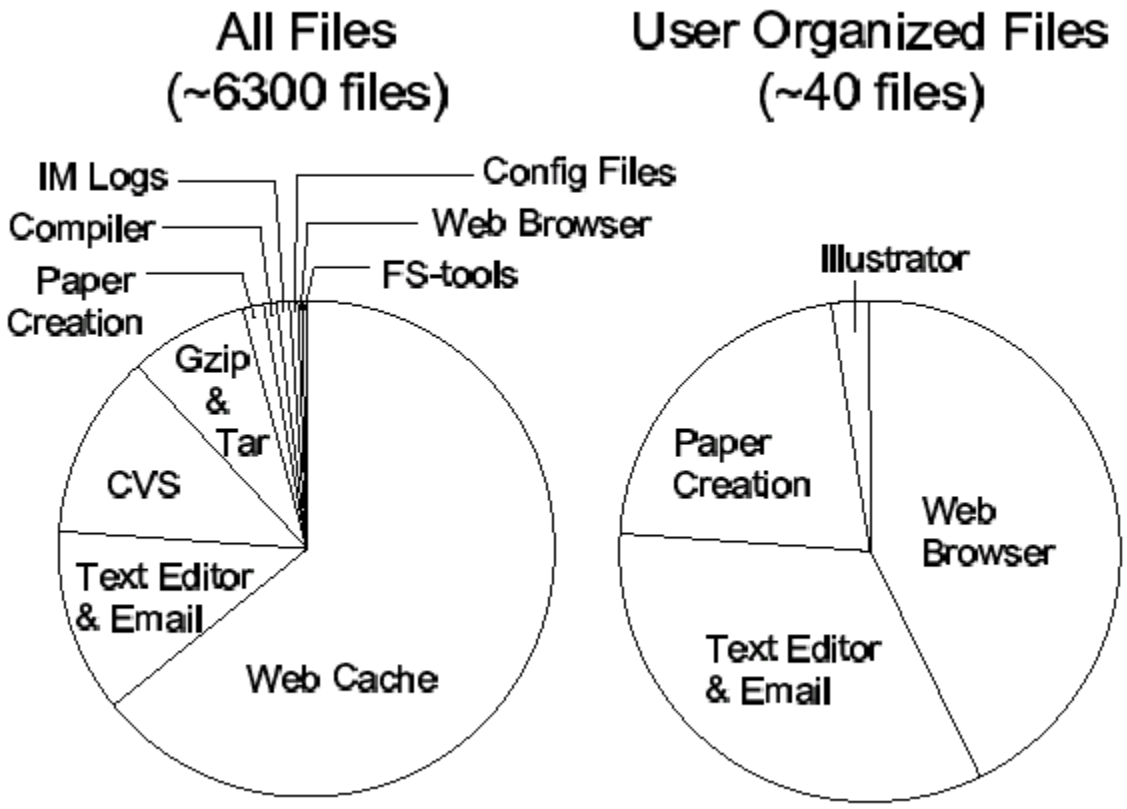
Context-based Relationships

- Premise: Context is what user might remember best.
- Previous work
  - Hoarding for disconnected access (inter-file relationships)
  - Google: textual context for link and feedback from search behavior (assumption of popularity over many users)

Access-based

- Use context of user's session at access time
- Application knowledge – modify apps to provide hints
  - Example: subject of email associated with attached file
- Feedback from "find" type queries
  - Searches are for rarely accessed files and usually only one user – limits statistical info

Traced File Creation Behavior



Inter-file

- Attributes can be shared/propagated among related files
- Determining relationships
  - User access patterns – temporal locality
  - Inter-file content analysis
    - Similarity – duplication -- hashing
    - Versions

Challenges

- Evaluation – issues of deployment for user study
- Mechanisms
  - Storage of large numbers of attributes that get automatically generated
  - User interface
- Context switches
  - Creating false positive relationships

Background:
Inter-file Relationships

Hoarding - Prefetching for Disconnected Information Access

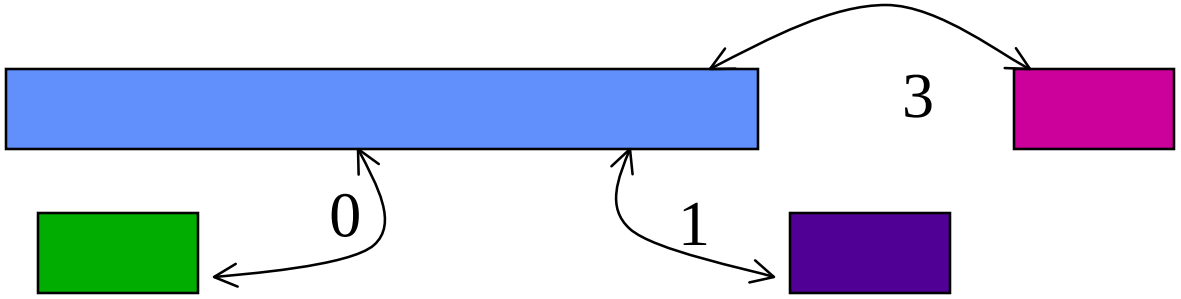
- Caching for availability (not just latency)
- Cache misses, when operating disconnected, have no redeeming value.
  (Unlike in connected mode, they can't be used as the triggering mechanism for filling the cache.)
- How to preload the cache for subsequent disconnection? Planned or unplanned.
- What does it mean for replacement?

SEER's Hoarding Scheme: Semantic Distance

- Observer monitors user access patterns, classifying each access by type.
- Correlator calculates semantic distance among files
- Clustering algorithm assign each file to one or more projects
- Only entire projects are hoarded.

Defining Semantic Distance

- Temporal semantic distance - elapsed time between two file references
  Time scale effects :-(
- Sequence-based semantic distance - number of intervening file references between 2, of interest. At what point? Open? Close?
- Lifetime semantic distance - accounts for concurrently open files - overlapping lifetimes



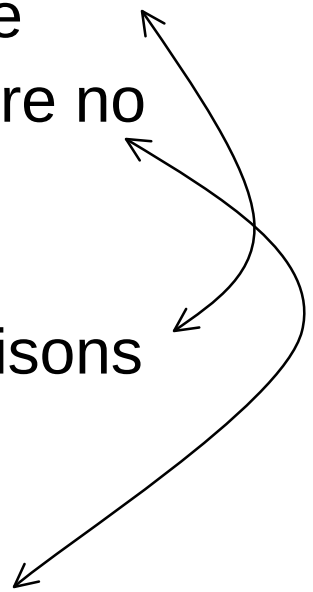
- How to turn semantic distance between two references into semantic distance between files? Summarize - geometric mean.
- Using months of data. Only store n nearest neighbors for each file and files within distance M
- *External investigators* can incorporate some extra info (e.g. heuristics used by Tait, makefile)

Real World Complications

- Meaningless clutter in the reference stream (e.g. find command)
- Shared libraries - an apparent link between unrelated files - want to hoard but not use in distance calculations and clustering
- Rare but critical files, temp files, directories
- Multi-tasking clutter
- Delete and recreate by same filename.
- Examine metadata then open – 1 or 2 accesses?
- SEER tracing itself – avoid accesses by root

Evaluation

- Metric
  - Hoard misses usually do not allow continuation of activity (stops trace) – counting misses is meaningless.
  - Time to 1st miss – would depend on hoard size
  - Miss-free hoard size – size necessary to ensure no misses
- Method
  - Live deployment – difficulty in making comparisons
    - Only long enough disconnections
    - Subtract off suspensions
  - Trace-driven simulation -- reproducible
    - What kind of traces are valid?



Context of Hoarding

Disconnected and Weakly Connected Coda

Satya, Kistler, Mummert, Ebling, Kumar, and Lu, “Experience with Disconnected Operation in a Mobile Computing Environment”, *USENIX Symp. On Mobile and Location-Independent Computing*, 1993.

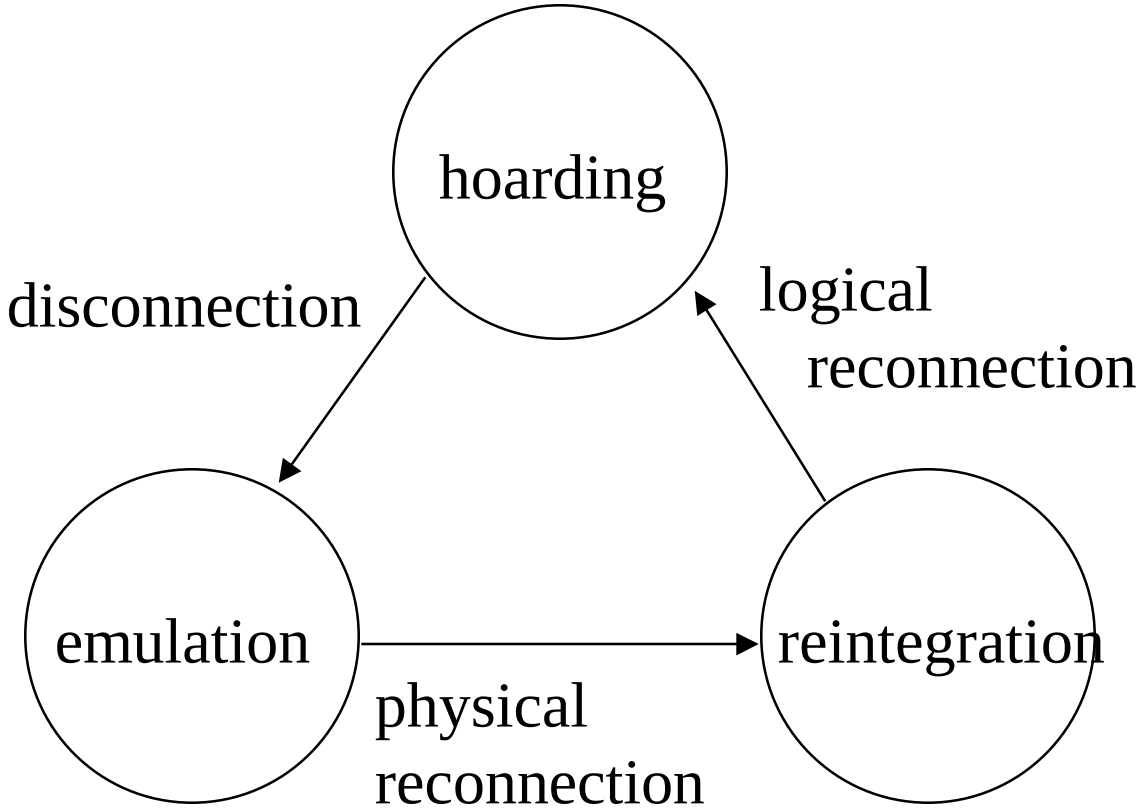
Mummert, Ebling, and Satya, “Exploiting Weak Connectivitiy for Mobile File Access”, *SOSP95*.

Coda

- Single location-transparent UNIX FS.
- Scalability - coarse granularity (whole-file caching, volume management)
- First class (server) replication and second class (client caching) replication
- Optimistic replication & consistency maintenance.

→ Designed for disconnected operation for mobile computing clients

Client-cache State Transitions



Hoard Database

- Per-workstation, per-user set of pathnames with priority
- User can explicit tailor HDB using scripts called *hoard profiles*
- Delimited observations of reference behavior (snapshot spying with bookends)

Coda Hoarding State

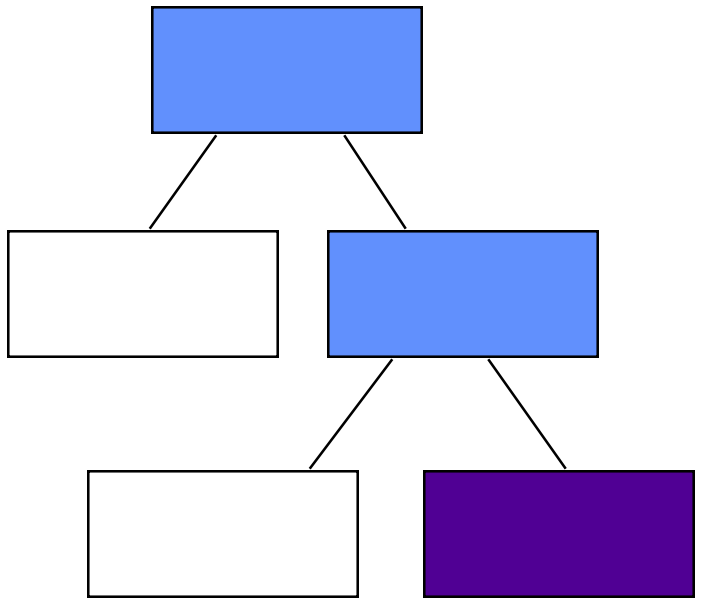
- Balancing act - caching for 2 purposes at once:
  - performance of current accesses,
  - availability of future disconnected access.
- Prioritized algorithm -
  Priority of object for retention in cache is
  f(hoard priority, recent usage).
- Hoard walking (periodically or on request) maintains equilibrium - no uncached object has higher priority than any of cached objects

The Hoard Walk

- Hoard walk - phase 1 - reevaluate name bindings (e.g., any new children created by other clients?)
- Hoard walk - phase 2 - recalculate priorities in cache and in HDB, evict and fetch to restore equilibrium

Hierarchical Cache Mgt

- Ancestors of a cached object must be cached in order to resolve pathname.
- Directories with cached children are assigned infinite priority



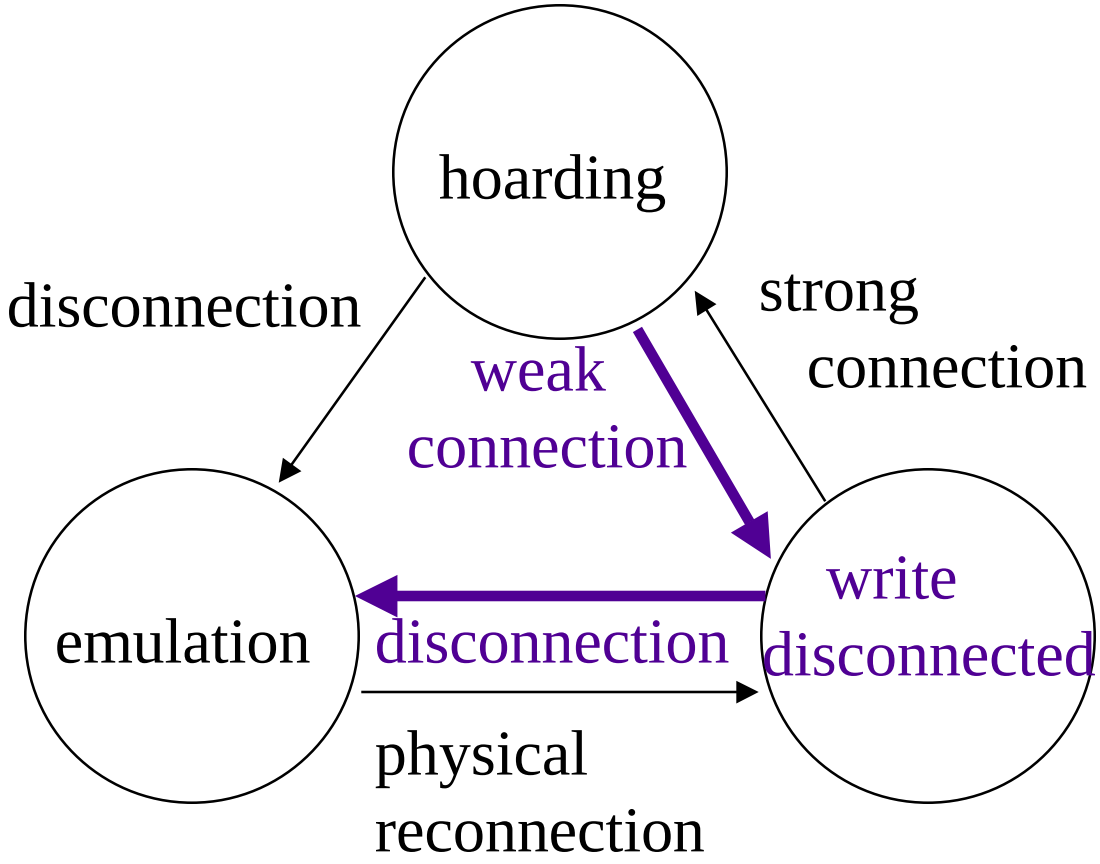
Callbacks During Hoarding

- Traditional callbacks - invalidate object and refetch on demand
- With threat of disconnection
  - Purge files and refetch on demand or hoard walk
  - Directories - mark as stale and fix on reference or hoard walk, available until then just in case.

Emulation State

- Pseudo-server, subject to validation upon reconnection
- Cache management by priority
  - modified objects assigned infinite priority
  - freeing up disk space - compression, replacement to floppy, backout updates
- Replay log also occupies non-volatile storage (RVM - recoverable virtual memory)

Client-cache State Transitions with Weak Connectivity



Cache Misses with Weak Connectivity

- At least now it's possible to service misses but $$$ and it's a foreground activity (noticable impact). Maybe **not**
- User patience threshold - estimated service time compared with what is acceptable
- Defer misses by adding to HDB and letting hoard walk deal with it
- User interaction during hoard walk.