Lee Beckman
April 15, 2012

Thesis Analysis/Evaluation/Writing Plan

Current status:

- -AspectJ weaving of complete set of classes (including pre-compiled JSPs) is working.
    - -The main potential issue is dependence of array access (find workarounds as needed)

Goal:

- -Demonstrate that the graphs I am now producing (the results of taint tracking) are useful
    - -Produce good graphs

High level steps:

- -Ensure reasonable completeness of tracker
    - -Must be able to track the flow of data (at least in strings) completely enough (from beginning to all endpoints) so that we can perform useful analyses later.
    - -Must be able to track enough actual data (taint sources) to perform useful analyses later.

- -Analyses to perform
    - -Basic state location
    - -Fluxo-style (caching, pre/post comp)
    - -Moving state (derived state, relocating state)
    - -Access path refactoring

- -Evaluation
    - -Toy applications will be developed alongside analyses
    - -For toy application, can go with the simple web store application from my earlier slides
    - -Select a single real application to analyze and modify

Lee Beckman
April 15, 2012

Week-by-week:

Analysis/Light Evaluation

April 16	**Tracker Evaluation / Data Collection** Collect traces of JGossip data for later analysis, and to continue evaluating the completeness of the tracker. Driving JGossip, and any web application, should be as simple as having a script of web requests, or using something like selenium for automation. This way I can easily produce the full trace after modifying the tracking code. Rounding out the tracker will involve graphing the collected data, and comparing it with what I know about how the application works, ensuring the flows of various pieces of data have reasonable start and end points. This process has already begun. Simple source code inspection will be a big part of this process. This could involve testing another, potentially simpler application.
April 23	**Data Collection of Another Application / Analysis Planning** Select an application (JForum, Trading application, Simple web store, etc) Drive this application as with JGossip, collect data for analysis Use the collected data to estimate which analyses will be important to focus on For example, I may see lots of state, requiring means to organize how I present it to the developer, or places where I might miss state and need to develop more sophisticated analyses to catch it Create a rough plan for developing semi-realistic toy-applications to demonstrate the ability of my tool.
April 30	**Analysis: State Finding, Access Path Refactoring** **Create toy application:** Keep session state in memory Have personal / shared state Store state in database Access multiple data sources together in ways both highly coupled and not **Example:** Online store which maintains a simple shopping cart for each user (personal state) Also keep state to share site activity between users, such as who is viewing what (shared state) Have special items which are loaded from a separate data source along with items (non-coupled data, could modify access path without affecting how items data is accessed) Have stock data used in calculating which items to display (couple data – relocating it would affect how items data is accessed)

	Apply to real applications **Output:** Class/Method/Variables/Tables responsible for keeping state Where the state originates from Indication of path state takes through graph Whether or not state is personal Data source dependencies Points in paths where data is coupled **Potential Problems** Real applications may keep very little in-memory state, opting instead to put everything in the database Every data source may be accessed separately, offering little interesting to say about access path refactoring, or Data sources may be coupled in such complex ways that it is very difficult to suggest how they could be accessed independently
May 7	**Analysis: Caching, Pre/Post Computation** **Create toy application:** Perform computation-heavy processing on data before it reaches the user Some computation should involve unpredictable sources Could indicate sources to developer and ask about predictability Perform computation which could be deferred (outputs not sent to user immediately) Perform processing on static, unchanging data **Example:** Online store can generate various pages. Some of these could be simply pre-computed outright, others could have small subcomponents which could be stored in caches After a user buys something, we can perform some computation to generate a receipt to be emailed, which could be moved to post-computation **Apply to real applications** **Output:** Points in graph where data used is transformed/used in computation Computations with potentially predictable inputs Computations with unpredictable inputs Potential cache location suggestions Output computations which do not reach the user in a request **Potential Problems** Finding good places to cache may require some additional profiling data such as execution costs. This may be a better problem for later tools consuming my data Distinguishing between caching and pre-computation opportunities Opportunities for post-computation could be rare

May 14	**Analysis: State Relocation/Elimination (opposite of caching?)** **Create toy application:** Have personal state. Create state which can be derived from existing state given some computation **Example:** Shopping cart state could be moved to the user This state can involve some computation which formats the data for display **Apply to real applications** **Output:** State which could potentially be moved to the client Computation which could be moved to client as well, assuming state moved Suggestions for state elimination **Potential Problems:** May be very difficult to determine what is personal state Suggestions to cache computation (create state) vs eliminate state by performing more computation may seem conflicting

Lee Beckman
April 15, 2012

Extended Evaluation

May 21	Expand toy examples More complexity may be needed to be convincing May discover further interesting analyses to demonstrate
May 28	Modify JForum/JGossip/etc Will have a better idea of which one to use later. Having applied analyses to real applications while developing them, I can hopefully select a good example which exhibits many of the characteristics I am looking for Should already have graphs of these applications collected by this point
June 4	Continue modifying real application The process could simply be to show the correctness of the results Would also like to modify the application (add caches, move state to user, change how data sources are accessed), show that the application still works, and then explain how this would be useful in such scenarios as partitioning.
June 11	Integration with Nima / Short User Study? User-study was an idea given during my presentation to the MAGIC lab Would be interesting to see how other developers interpret the data my tool provides
June 18	Overflow / Extended Evaluation
June 25	Overflow / Extended Evaluation

Lee Beckman
April 15, 2012

Writing

July 2	Writing
July 9	Writing
July 16	Writing
July 23	Writing

Lee Beckman
April 15, 2012

Finalizing

July 30	Revisions
August 6	Revisions / Defense Preparation
August 13	Defense Preparation
August 20	Defense