Formal dependency management

“Results from EDOS WP2”

Oğuz Berke DURAK

`please@spam.me.to.death`

INRIA Rocquencourt, The EDOS Project

Formalizing dependencies

- Package management is very complex
- It has real, non-obvious algorithmic problems
  - How to find broken packages ?
  - How to migrate packages from unstable to testing ?
  - How to select packages to build a useful set of DVDs ?
- We must simplify and formalize to understand those problems.

Boolean logic

- Suitable for representing dependency constraints
- Variables represent packages
- An installation is a valuation: a variable is true when the package is present in the installation.
- Dependency is implication : $p \rightarrow q$ means that p depends on q
- Conflicts are given by a disjunction of negated literals : $\neg p \vee \neg q$ means that p and q conflict.
- In fact, we do have disjunctive dependencies because of multiple versions or the “provides” mechanism.

A giant Boolean formula

- The conjunction of all the dependency constraints for all the packages in an archive gives a big formula F
- F is of the form :

$$F = \bigwedge_{p \text{ and } q \text{ conflict}} (\neg p \vee \neg q) \wedge \bigwedge_{p \text{ depends on } q_1 \text{ or } q_2 \ldots} (p \rightarrow q_1 \vee \cdots q_k)$$

- For Debian i386, F has about 370,000 clauses (about 10 per package)
- A package p is installable if and only if the formula $F \wedge p$ is satisfiable.
- A package that is not installable is *broken*.
- In fact, package installability is an NP-complete problem

The complexity of installability

- NP-completeness is not very relevant
- However, the problem is not easy
  - Simple backtracking won't work for many packages
  - Davis-Putnam SAT-solving takes too much time (tens of minutes) on some difficult packages (abiword, achims-guestbook...)
  - Similarly, standard search strategies in CLP languages (Oz) may take too much time
  - APT has heuristics that work most of the time but fails on some real instances
  - Smart may take many months (!) for some packages

EDOS Contributions – 1

- Tools for parsing, storing, visualizing, converting and browsing package metadata
- `ceve` (Jaap BOENDER, OCaml): generic metadata converter (handles Debian and RPM formats)
- `edos-toolchain` (Fabio MANCINELLI, Java): dependency encoder, visualizer
- `anla` (Berke DURAK, OCaml): metadata browser

EDOS Contribution – 2 (J. Vouillon)

- An empirically efficient algorithm for solving installability
- Implemented `debcheck/rpmcheck`, `anla` and in production (`http://brion.inria.fr/anla/` Debian QA, Caixa Magica)
- Can check a whole repository (40,000 packages) in two minutes.

The thinning problem

- We want to build a set of DVDs for our distribution
- We have a limited number of DVDs
- We only want to put the best stuff
- The DVDs should be self contained (w.r.t. dependencies): no broken packages
- The DVDs must be ordered by dependency (the first DVD is self-contained; the second DVD may depend on itself and the first one...)

EDOS Contribution – 3

- An empirically efficient algorithm for thinning
- Devised by myself by refining a simple backtracking dependency solver
- Works really well and is fast
- Implemented in `tart` (OCaml)
- Can also be used for installability checking
- About as fast as the other algorithm

Scheduled work

- The EDOS project is ending in three months
- The community must take over the work done
- Linux distributions need to support a common initiative to build and develop formal dependency management tools

Things to do

- Develop pkglab
- Migration process
- Upgradeability tests
- Checking for loss of functionality