Тезисы доклада:	Deepsolver — инструмент управления пакетами с программным обеспечением
Автор:	Михаил Пожидаев
Конференция:	Девятая конференция разработчиков свободных программ
Место проведения:	г. Обнинск
Дата:	23 июля 2012 г.

Deepsolver — новый инструмент для управления пакетами с программным обеспечением (ПО). Разработка ведётся сотрудниками компании "Альт Линукс" с целью замены менеджера *APT* новым продуктом, удовлетворяющим современным требованиям. Основные функции *Deepsolver*:

- доставка с удалённых узлов, установка, обновление и удаление пакетов;
- наполнение автоматизированной среды для компиляции исходных текстов;
- поиск доступных пакетов в удалённых репозиториях и предоставление информации о них;
- поддержание целостности операционной системы (ОС).

Разработка ведётся с выделением трёх основных компонент:

1. Обработка запросов пользователя на внесение изменений в ОС.
2. Поиск информации о доступных пакетах.
3. Построение индексов репозиториев.

Схема зависимостей и конфликтов между пакетами в современных дистрибутивах *GNU/Linux* может быть сведена к булевому уравнению. Это доказывает, что задача обработки запросов пользователя является задачей из класса *NP*-полных алгоритмов. Нахождение точного решения за полиномиальное время невозможно, и необходимо использовать методы приближённого или ограниченного поиска. Приближённый метод подразумевает сохранение задачи *NP*-полной с привлечением одного из известных приближённых алгоритмов решения булевых уравнений (*SAT*). Время вычислений сокращается, но нахождение решения не гарантируется даже в случае его существования. Ограниченный подход заключается в использовании точного полиномиального алгоритма с предварительным наложением дополнительных условий в формулировке задачи.

Архитектура *Deepsolver* допускает существование нескольких реализаций алгоритма одновременно. Тем не менее, реализация по умолчанию основана на ограниченном подходе. Это достигается путём наложения дополнительного ограничения "одна зависимость — одно возможное разрешение". Другими словами, сформулирован точный алгоритм, однозначно выбирающий пакет, подходящий под требования некоторой зависимости. Хотя в действительности за счёт существования записей *provides* подобных вариантов может быть несколько. Помимо возможности применять полиномиальный алгоритм, новое ограничение повышает предсказуемость работы в условиях формирования автоматизированного сборочного окружения.

Во время вычисления изменений состояния ОС в ответ на запрос пользователя обрабатываются три множества пакетов:

1. Пакеты, непосредственно указанные пользователем.

2. Пакеты, установка, обновление или удаление которых однозначно необходима вследствие прямых зависимостей или конфликтов пакетов, указанных пользователем.

3. Пакеты, установка, обновление или удаление которых необходима для сохранения целостности ОС.

Первые два множества пакетов в большинстве случаев могут быть вычислены однозначно, но при поиске путей сохранения целостности ОС возможно существование нескольких решений. Особое внимание вызывают пакеты, для которых принято решение об удалении или обновлении. Для них существует три варианта действий:

1. При нарушении зависимости на пакет в случае его обновления произвести обновление зависимых пакетов.

2. Произвести замену пакета, используя правила обработки зависимостей, если результат вычислений не будет совпадать с удаляемым или обновляемым пакетом.

3. Полностью удалить зависимое поддерево установленных пакетов.

Обновление зависимых пакетов имеет смысл выполнять только в случае обновления исходного пакета и должно рассматриваться отдельно. Для выбора между второй и третьей альтернативой можно использовать ограниченное решение булевого уравнения, в котором каждый элемент нормальной формы имеет только две переменных, т. е. так называемый *2-SAT*.

Пусть пакет p зависит от r_1 и r_2, а конфликтует с c_1 и c_2. В таком случае соответствующий фрагмент уравнения должен быть следующим: $(!p \vee r_1) \wedge (!p \vee r_2) \wedge (!p \vee !c_1) \wedge (!p \vee !c_2)$. В роли p выступают все пакеты, для которых необходимо принять решение, а также связанные с ними путём зависимостей или конфликтов. Для каждой альтернативы удаления поддерева или установки замещения в уравнение должен добавляться новый элемент вида $(p_1 \vee !p_2)$, где p_1 — пакет замещения, а p_2 — пакет удаления.

На первый взгляд кажется, что уравнение принимает огромные размеры, но его решение может быть найдено за линейное время. Каждый элемент вида $a \vee b$ эквивалентен двум импликациям: $!a \to b$ и $!b \to a$. Это позволяет совершить переход к ориентированному графу импликаций, в котором каждая переменная заменяется двумя вершинами: для положительного и для отрицательного значения. Справедливо утверждение, что решение существует, если не существует одновременно пути на графе из a в $!a$ и наоборот для любых a. Фактически, необходимым и достаточным условием для этого является требование, чтобы a и $!a$ принадлежали бы разным компонентам сильной связности. Их поиск может выполняться за линейное время и позволяет определить конкретное решение уравнения.