Тезисы доклада:	Deepsolver: статус разработки и предложения
Автор:	Михаил Пожидаев
Конференция:	X конференция разработчиков свободных программ
Место проведения:	г. Калуга
Дата:	21 сентября 2013 г.

Настоящая статья содержит обзор процесса разработки менеджера пакетов *Deepsolver*, описание некоторых трудностей, которые были встречены в ходе работы, и ряд предложений для дальнейшего развития проекта. В отличие от намерений, высказанных разработчиками *Deepsolver* ранее, нынешняя реализация утилиты основана на алгоритме *minisat*. Все попытки применить *2-SAT* алгоритмы приводили к существенному ограничению гибкости и по этой причине были отклонены. Поддержка *Deepsolver* интегрирована в набор утилит *girar* и функционирует на сборочной площадке *git.altlinux.org*.

Одна из самых сложных проблем, с которой пришлось столкнуться в процессе разработки, заключается в том, что *SAT*-решение затрагивает необоснованно большое число пакетов. На практике это приводит к удалению из системы пакетов, присутствие которых не делает выполнение задачи пользователя невозможным. Рассмотрим для примера фрагмент *SAT*-уравнения: $(!r \vee a \vee i)$. Здесь r — некоторый зависимый пакет, a— некоторый пакет доступный для установки из репозитория, i — некоторый пакет, установленный в системе. Если алгоритм выбрал для установки a, i может быть как “истина”, так и “ложь”, и это никак не влияет на выполнимость запроса пользователя. В математической постановке обе ветви решений имеют одинаковый приоритет.

В настоящий момент *Deepsolver* содержит две реализованные оптимизации, но их недостаточно. Первая из них пытается по мере построения уравнения предугадать, какие переменные получат достоверно то или иное значение, после чего исключает из уравнения клозы, выполнимость которых оказывается очевидной. После исключения количество затронутых пакетов уменьшается, и это даёт частичный желаемый результат.

Другая идея учитывает пакеты, которые уже установлены в системе. Если установленный пакет появляется в некотором клозе без отрицания, но больше нигде не появляется в уравнении, то пакет должен остаться установленным, а весь клоз с ним считается истинным. Этот метод в терминологии *Deepsolver* назван “отложенными клозами”, т. к. они добавляются в уравнение только после второго включения установленного пакета.

Предложенные методы частично решают описанные проблемы. К недостатку первого подхода, например, относится то, что он чувствителен к порядку вычислений. Они получили своё развитие в идее, которая описана ниже и находится сейчас в стадии реализации для постановки вычислительного эксперимента. Все клозы могут быть разбиты на группы, которые отражают действия, необходимые для возможности установки или удаления некоторого пакета. Между подобными группами может быть определено множество ссылок, которые отражают, какая группа приводит к необходимости обработки возможности установки или удаления пакета. Ссылки подсчитываются и после удаления некоторых из них запускается алгоритм “сборки мусора”. Удаление ссылок возможно как в случае удаления установленных пакетов, которые нигде не включены с отрицанием, до запуска *minisat*, так и после него для тех переменных, значение которых не изменилось.

Ниже предлагаются несколько дополнительных идей для дальнейшего развития *Deepsolver*. Все они подразумевают введение третьего уровня управления пакетами без изменения текущей функциональности (первым уровнем считается *librpm*). Пользователи, привыкшие к поведению утилит в стиле *APT*, не заметят отличий в работе.

Первое предложение заключается в добавлении новых сущностей для обработки пакет-

ным менеджером, таких как шрифты, jar-файлы и т. д. Внутренняя их поддержка должна быть реализована в гибкой форме, чтобы любое требуемое расширение их множества могло быть выполнено без модификации *C/C++* кода. Любая информация, необходимая для сопоставления пакетов и сущностей в них, должна подготавливаться большей частью автоматически на этапе сборки пакета, не запрещая мэйнтейнеру вносить корректировки по его желанию.

Главная причина, по которой такие функции возложены именно на пакетный менеджер, в том, что только он в системе имеет полную информацию о приложениях и данных, которые не только уже были установлены пользователем, но и могут быть установлены из репозитория. В случае рассинхронизации расширенной информации вывод пакетного менеджера останется всегда корректным, но, возможно, будет неполным. Необходимо быть осторожным, выбирая, какие типы новых сущностей должны быть добавлены. Неясно, стоит ли таким образом обрабатывать, например, модули ядра. Вероятно, часть функций администрирования должны производиться по-старому.

Другое предложение подразумевает отслеживание пакетов, которые меняют поведение системы. Например, к ним относится пакет *acpid-events-power*. Важной особенностью этих пакетов является то, что при их установке или удалении не должны затрагиваться никакие другие пакеты. Оба предложения, очевидно, хорошо подходят для того, чтобы повысить удобство управления системой при помощи оконных утилит администрирования.