УДК 004.021:519.857

ИССЛЕДОВАНИЕ ПРАКТИЧЕСКОЙ ЭФФЕКТИВНОСТИ БИТОВОЙ ОПТИМИЗАЦИИ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ В ИНФОРМАЦИОННО-КОМПЬЮТЕРНЫХ СИСТЕМАХ НА ПРИМЕРЕ ЗАДАЧИ О РЮКЗАКЕ 0-1

Марковец Р.С.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

Научный руководитель: Мигалевич С.А. – магистр техн. наук, ст. преподаватель кафедры информатики

Аннотация. Экспериментально исследована практическая эффективность битовой оптимизации динамического программирования для задачи о рюкзаке 0-1 в контексте вычислительных модулей информационно-компьютерных систем. Установлено, что реальное ускорение составляет от 15,6 до 21,0 раза в зависимости от распределения весов предметов, что в 3-4 раза ниже теоретического предела ускорения в 64 раза. Выявлены основные факторы расхождения: кэш-промахи при обработке битового множества, накладные расходы библиотечной реализации и межсловные сдвиги.

Ключевые слова: битовое множество, задача о рюкзаке, оптимизация алгоритмов, практическая эффективность, спортивное программирование, информационно-компьютерные системы, вычислительные модули

Введение. Битовая оптимизация динамического программирования (далее – ДП) на основе битовых множеств применяется в информационно-компьютерных системах реального времени [1]:

– в логистике – для решения задачи о рюкзаке 0-1 (далее – задача о рюкзаке);

– в телекоммуникациях – при распределении каналов;

– в криптоанализе;

– во встраиваемых системах и сетевых устройствах с ограниченными вычислительными ресурсами.

Повсеместно требуется обрабатывать десятки тысяч предметов за миллисекунды, и битовая оптимизация позволяет уложиться в жесткие временные рамки. Однако на практике теоретический выигрыш в 64 раза недостижим из-за факторов, не учитываемых асимптотической моделью.

При проектировании вычислительных модулей опираться только на оценку ускорения в 64 раза недостаточно. Реальная производительность ограничена задержками памяти, кэш-промахами, накладными расходами языковых конструкций и свойствами входных данных. Предсказать ускорение без эксперимента невозможно, поэтому цель работы – количественно оценить фактическую эффективность битовой оптимизации.

В исследовании планируется сравнить наивное ДП на булевом массиве и оптимизированную версию с *bitset* на различных наборах весов. Будут измерены ускорение и проведен анализ ограничивающих факторов. Результаты позволят разработчикам информационно-компьютерных систем точнее прогнозировать эффект от данной оптимизации.

Основная часть. Рассматривается классическая задача о рюкзаке [2]. Имеется множество из n предметов, каждый предмет i характеризуется целым неотрицательным весом w_i. Вместимость рюкзака характеризуется целым неотрицательным числом W. Требуется выбрать подмножество предметов так, чтобы их суммарный вес не превышал W и был максимален.

Наивный метод ДП использует булев массив dp размером $W+1$, где элемент с индексом s показывает, достижима ли сумма s. Начальное состояние: достижима только нулевая сумма. Для каждого предмета весом w массив обновляется в порядке убывания s от W до w по формуле (1):

$$dp[s] := dp[s]\ OR\ dp[s - w], \quad (1)$$

где *OR* – логическое ИЛИ.

После обработки всех предметов ответом служит наибольший индекс *s*, для которого значение истинно. Временная сложность такого решения составляет *O(nW)*, память – *O(W)*.

Булев массив можно заменить битовым множеством (*bitset*) [3]. В такой структуре каждый бит соответствует одной сумме: если бит установлен, сумма достижима. Битовое множество хранится как последовательность машинных слов, что позволяет выполнять логические операции и сдвиги сразу над группами битов. Добавление предмета весом *w* сводится к формуле (2):

$$B := B\ OR\ (B \ll w), \quad (2)$$

где *B* – текущее битовое множество;

$\ll w$ – побитовый сдвиг влево на *w* позиций.

Эта операция одновременно добавляет все суммы, увеличенные на *w*, к уже достигнутым. Современные 64-разрядные процессоры обрабатывают 64 бита за один такт, что в идеальных условиях должно давать ускорение в 64 раза по сравнению с наивной версией. Однако на практике достичь такого ускорения мешают ограничения аппаратного обеспечения и накладные расходы, связанные с реализацией битовых множеств.

Экспериментальное исследование проводилось на ноутбуке с 64-разрядным процессором *Intel* тактовой частотой около 3 ГГц. Выбор платформы не является определяющим: относительное ускорение, достигаемое за счет битовой оптимизации, слабо зависит от конкретной модели процессора, поскольку определяется прежде всего разрядностью архитектуры и общими принципами организации памяти, едиными для подавляющего большинства вычислительных устройств, применяемых в информационно-компьютерных системах. Таким образом, полученные результаты репрезентативны для широкого класса платформ.

Программная реализация выполнена на языке *C++*. Для наивного ДП применялся булев массив на основе специализации *std::vector<bool>*. В оптимизированной версии булев массив заменен битовым множеством фиксированного размера. Поскольку вместимость рюкзака принята равной $W = 10^6$, использовалось битовое множество размером *W+1 = 1 000 001* бит (в синтаксисе языка – *std::bitset<1000001>*). Операции сдвига и логического ИЛИ над битовым множеством реализованы встроенными средствами библиотеки. Время выполнения измерялось с помощью *std::chrono::high_resolution_clock*. Каждый эксперимент повторялся пять раз для учета случайных отклонений.

Генерация входных данных выполнялась для трех распределений весов: равномерного, геометрического и гармонического. Выбор обусловлен необходимостью оценить влияние структуры набора предметов на ускорение. Худшие случаи на практике редки, тогда как выбранные распределения охватывают широкий спектр реальных ситуаций. Для каждого распределения генерировалось по 1000 предметов.

Полученные отношения времени выполнения наивной реализации ко времени работы с битовым множеством приведены в таблице 1:

Таблица 1 – Отношение времени выполнения алгоритмов, мс

Распределение	Запуск 1	Запуск 2	Запуск 3	Запуск 4	Запуск 5	Среднее
Равномерное	16.1	16.6	16.5	17.1	15.6	16.4
Геометрическое	15.3	15.4	15.4	18.5	13.7	15.6
Гармоническое	20.6	21.3	21.1	20.6	21.3	21.0

Выявленное расхождение между теоретическим ускорением в 64 раза и полученными на практике 16-21 разом объясняется совокупностью нескольких факторов, например:

– ограничения, связанные с иерархией памяти: битовое множество размером 125 КБ не помещается в кэш первого уровня, что вызывает задержки при обращениях к памяти;

– накладные расходы реализации битового множества в стандартной библиотеке: дополнительные инструкции для проверки границ и циклической обработки слов;

– межсловные сдвиги: при сдвигах, не кратных 64, возникают переносы битов между машинными словами, требующие дополнительных операций.

Заключение. В ходе экспериментального исследования установлено, что битовая оптимизация ДП для задачи о рюкзаке на основе битовых множеств обеспечивает реальное ускорение от 15,6 до 21,0 раза в зависимости от распределения весов предметов. Это значительно ниже теоретического предела в 64 раза. Основными причинами расхождения являются кэш-промахи при обработке битового множества, которое не помещается в кэш первого уровня, накладные расходы библиотечной реализации, а также дополнительные операции при межсловных сдвигах.

Полученные результаты свидетельствуют о критическом расхождении между теоретической и практической эффективностью: реальный выигрыш в 3-4 раза ниже ожидаемого. При проектировании вычислительных модулей информационно-компьютерных систем опора исключительно на теоретическую оценку ускорения может привести к завышенным ожиданиям и, как следствие, к ошибочному выбору методов оптимизации.

Представленные количественные оценки могут быть использованы разработчиками информационно-компьютерных систем в качестве ориентира при приближенной оценке эффекта от внедрения битовой оптимизации в задачах дискретной оптимизации, подобных задаче о рюкзаке. Результаты работы также подчеркивают важность экспериментальной валидации алгоритмических оптимизаций в контексте реальной вычислительной среды.

Список литературы

1 *Окулов С. М., Пестов О. А. Динамическое программирование / С. М. Окулов, О. А. Пестов. М.: БИНОМ. Лаборатория знаний, 2018. 296 с.*

2. *Kellerer H., Pferschy U., Pisinger D. Knapsack Problems / H. Kellerer, U. Pferschy, D. Pisinger. 1st ed. Berlin, Heidelberg: Springer, 2004. 548 p.*

3. *bitset Class Template Reference [Электронный ресурс]. Режим доступа: https://gcc.gnu.org/onlinedocs/libstdc++/libstdc++-html-USERS-4.1/classstd_1_1bitset.html. Дата доступа: 12.02.2026.*

UDC 004.021:519.857

OPTIMIZATION OF COMPUTATIONAL MODULES IN INFORMATION AND COMPUTER SYSTEMS: A CASE STUDY OF THE KNAPSACK PROBLEM

Markovets R.S.

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Migalevich S.A. – Master of Tech. Sci, senior lecturer at the department of computer science

Annotation. The practical efficiency of bit optimization of dynamic programming for the 0-1 knapsack problem in the context of computational modules of information and computer systems has been experimentally studied. The actual speedup is found to range from 15.6 to 21.0 times depending on the item weight distribution, which is 3-4 times lower than the theoretical speedup limit of 64 times. The main factors contributing to this discrepancy are identified: cache misses during bitset processing, overhead of the library implementation, and cross-word shifts.

Keywords: bitset, knapsack problem, algorithm optimization, practical efficiency, competitive programming, information-computer systems, computational modules