Анализ на задача 5 от конкурса на Мусала Софт и PC Magazine, 2006г. (“Списъци”)

Приготвен от Веселин Георгиев

При решаването на задачата целта беше по-скоро простота и нисък константен фактор, отколкото ниска алгоритмична сложност. Решението реализира O(√N) сложност за повечето операции (където N е дължината на списъка, върху който е приложена операцията).

Структурата данни, която се използва, е двойносвързан списък. Елементите в него са масивчета от по 512 елемента. Тези масиви са фрагменти от целия “списък” (по терминологията на условието). В двойносвързания списък освен тези масивчета се пази допълнително следната информация (за всеки елемент):

- Min и Max стойност от масива;
- Direction – определя как елементите са подредени в действителност (идеята е, че при “reverse” команда, просто се обхождат всички елементи от двойносвързания списък и да им се сменя direction-а);
- Брой нефиктивни елементи (ще изясним това по-долу).

Ето и как се реализират изискванията в условието операции:

- min – обхожда се целия двойносвързан списък с O(N/512) сложност и се разглежда “Min” полето на всеки елемент;
- max – по аналогичен начин;
- reverse – обхожда се целия двойносвързан списък с O(N/512) сложност и се сменят флаговете за посока, както и се пренареждат връзките между самите елементи на двойносвързаната структура;
- create – Създава се нов двойносвързан списък с единствен масив в него, в който има само едно число. Сложността тук е 512 операции, тъй като масива се зачиства;
- merge – Връзва се края на единия двойносвързан списък с началото на другия. Това може да стане с O(1) сложност, но е направено с O(512) по причини, които ще изясним по-долу;
- split – Търси се в кой елемент (масив) от двойносвързания списък се намира мястото на разделяне. Нека този масив обозначим с S. Търсенето става с O (N/512) сложност:
    - Сцепва се двойносвързания списък след S;
    - За новия списък се заделя ново масивче, което ще играе като начално и в него се прехвърлят част от елементите на масивчето на S. Прехвърлят се точно тези елементи, които няма да принадлежат на първия списък след сцепването. Това става с O(512) сложност.
    - Елементите в масивчето на S, които след цепенето няма да принадлежат на първия списък се маркират като фиктивни (присвоява им се стойност 0). Тук също така се работи и с полето “брой нефиктивни елементи” на масивчетата.
    - Поправят се стойностите на min/max полетата за S и за новото масивче.
    - Поправят се връзките на списъците.

Наблюдателния читател сигурно е забелязал, че при операция split се създават фиктивни елементи в началото и края на резултатните списъци. Ако ги merge-нем отново, ще се получи дупка от фиктивни елементи в средата на списъка. Тези дупки са лошо нещо; те вдигат сложността на всички операции. Взети са мерки за справяне с тях:

- При merge операция се гледа дали на мястото на сливане не се получава дупка с повече от 511 фиктивни елемента. В такъв случай се прави "кондензация" като елементите от две масивчета се вкарват в едно единствено. Това работи учудващо добре и решава проблема почти изцяло, като се и пести памет. Цената на кондензацията е сложност O(512) за merge операция.
- Въпреки всичко, последователност от много операции split и merge постепенно ще напълнят списъка с малки "неизбежни" дупки. Ето и какви мерки се вземат срещу тях: ако даден списък е достатъчно голям (SIZE_LIMIT) и отношението всички / нефиктивни елементи е достатъчно високо (ALL_NONFICTIVE_RATIO) се изпълнява "пълна кондензация": всички елементи на списъка се записват в обикновен масив, след което списъка се създава наново. Тежестта на пълната кондензация е O(N), но тя се среща доста рядко. Най-тежкия случай при официалните тестове е седмият (LISTS.7.INP), който поражда 15 кондензации. При тестове, генерирани от автора и специално целящи предизвикване на много кондензации, бройката достига максимум 60.
- В реализацията, пратена на журито, SIZE_LIMIT = 5000, ALL_NONFICTIVE_RATIO = 4

Ето и една допълнителна оптимизация, приложена с цел да се свали още константният фактор:

- Ако броят елементи е по-малък от една граница (около 512), вместо двойносвързан списък се ползва STL-ски vector. При него всички операции се правят буквално, т.е. min и max са с обхождане на всички числа, reverse с действително обръщане и т.н. така, че всички операции са с тежест O(512), но с много по-нисък константен фактор.

При писането на решението, автора е ползвал интензивно програмата valgrind (под Linux), помогнала му да открие няколко "тънки" бъга и да оправи два memory leak-а.