**УДК 519.6**

**О.М. БЕВЗ, А.С. ВАСЮРА, А.І. ЛІСОВЕНКО**

# ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ ДИНАМІЧНОГО РОЗПОДІЛЕННЯ ПАМ'ЯТІ В ОПТОЕЛЕКТРОННИХ СИСТЕМАХ

***Вінницький національний технічний університет***
***Хмельницьке шосе 95, м. Вінниця, Україна 21021***
***тел. +38(093)18-68-399, e-mail: sweettee@ukr.net***

**Анотація.** Необхідність застосування методів розподілу динамічної пам'яті в оптоелектронних системах з високими показниками ефективності є актуальною задачею. Сучасні програмні засоби оптоелектронних систем характеризуються великим часом роботи при реалізації методу розподілу динамічної пам'яті у випадках формування блоків пам'яті малого об'єму. В даній статті наведено метод зменшення часу роботи програмної реалізації методу розподілу динамічної пам'яті за рахунок використання нових структур даних управління пам'ятю в оптоелектронних системах – структура пошуку та структура склейки.

**Аннотация.** Необходимость применения методов распределения динамической памяти в оптоэлектронных системах с высокими показателями эффективности есть актуальной задачей. Современные программные средства оптоэлектронных систем характеризуются большими часовыми затратами при реализации метода распределения динамической памяти в случаях формирования блоков памяти маленького объема. В этой статье приведен метод уменьшения времени работы программной реализации метода распределения динамической памяти за счет использования новых структур данных управления памятью в оптоэлектронных системах – это структура поиска и структура склейки.

**Abstract.** Application a method of manage dynamic memory in the optoelectronic system is a actual goal. Contemporary software of the optoelectronic system has maximal time of a work in expirience forming block of the memory this minimal size. In this article given the method of decreasing time of the work when application new structure of a data in optoelectronic system – a structure of finding and a structure of concatination.

**Ключові слова.** Динамічне виділення пам'яті, оптоелектронна система, блок пам`яті, фрагментація, метод першого відповідного блоку, структура пошуку даних, структура склейки даних, операція пошуку найкращого підходящого.

## АКТУАЛЬНІСТЬ

Сучасні оптоелектронні пристрої представляють складні інтелектуальні системи до складу яких входять такі обчислювальні вузли, як пам'ять та процесор. Одним з типів пам'яті, яка використовується в таких системах є динамічна пам'ять. Ефективна робота динамічної пам'яті в оптоелектронних системах залежить від методів виділення та звільнення певних блоків фізичної пам'яті. Особливо різко постає проблема розподілу динамічної пам'яті в оптоелектронних системах при проектуванні оптоелектронних систем, коли необхідно виконувати обробку даних невеликого обсягу. Існуючі методи динамічного розподілу пам'яті є універсальними, і неефективно працюють при розміщенні малих програмних об`єктів. До того ж вони працюють досить повільно і займають великий об'єм пам'яті, що є недоліком при роботі. Формування методу виділення динамічної памяті, який має високі обчислювальні показники є актуальною задачею. Такими показниками є малий обсяг фізичної пам'яті оптоелектронної системи та невеликий час роботи алгоритму на основі методу формування динамічної пам'яті. Метою даної статті є формування методу динамічного розподілу пам'яті в оптоелектронних системах, який характеризується мінімізацію використання пам'яті, максимізацією сумісності, мінімізацією часових витрат та максимізацією мобільності.

## ПОСТАНОВКА ЗАДАЧІ

Динамічний розподіл пам'яті оптоелектронної системи – це управління ресурсами оптоелектронної системи під час виконання прогарами. За допомогою динамічного розподілу пам'яті можна гнучко керувати часом життя об'єктів оптоелектронної системи, що, на відміну від стекового методу розподілу, не призводить до неконтрольованого збільшення необхідної області пам'яті [1].

Динамічний розподіл пам'яті оптоелектронної системи поєднує в собі три методи:

1. виділення пам'яті для оптоелектронної системи, яка по суті є блоком пам'яті необхідного розміру, утвореного з блоків фізичної пам`яті;
2. звільнення блоку пам'яті, що більше не потрібний, для подальшого використання;
3. склеювання між собою блоків фізичної пам'яті оптоелектронної системи малого розміру для найбільш ефективного їх використання в майбутньому.

Існує ряд універсальних методів динамічного розподілу пам'яті, які застосовуються в сучасних оптоелектронний системах. Основними критичними параметрами цих методів, які обумовлюють ефективність розподілу пам`яті, є:

– час роботи центрального процесора оптоелектронної системи;

– обсяг пам'яті оптоелектронної системи, що потрібен для реалізації методу розподілу динамічної пам'яті;

– сумісність реалізацій (оптоелектронні системи мають різноманітний програмно-апаратний тип реалізації).

В відповідності до мети даної статті необхідно розробити метод розподілення динамічної пам`яті оптоелектронної системи, який має високу ефективність реалізації – зменшення часу роботи центрального процесора оптоелектронної системи, зменшення пам`яті для програмної реалізації методу розподілення пам`яті.

## ВИРІШЕННЯ ЗАДАЧІ

Рішення поставленої задачі буде виконано наступними кроками. На першому кроці буде проаналізовано сучасні методи, які використовуються для розподілу динамічної пам'яті в оптоелектронних системах. На другому – визначення компонентів, які впливають на ефективність роботи усього методу та формування методу розподілу динамічної пам'яті, який має високі показники ефективності – малий обсяг пам'яті та малий час роботи центрального процесора для реалізації даного методу в оптоелектронних системах в порівнянні з існуючими методами.

Методи динамічного розподілу пам'яті в сучасних оптоелектронних системах можуть бути згруповані в три категорії:

1. Методи послідовної відповідності: метод першого відповідного блоку, наступного відповідного і найкращого блоку [2, 3].
2. Методи, що використовують вільні виділені списки: просте виділене зберігання, виділений придатний, і т.п [4, 5].
3. Методи близнят: двійкові і подвоєні ієрархічні системи [6].

Проте, за умови невеликого об'єму даних оптоелектронної системи, ці методи мають певні недоліки. Наприклад, недоліком методу послідовної відповідності є значна фрагментація (більші блоки пам'яті на короткий час та менші – на довгий). У цьому випадку вивільнений великий блок буде, швидше за все, відразу використаний для виділення пам'яті відповідно до запиту на малий обсяг і після цього не вивільниться найближчим часом.

Недоліком методу, що використовує виділені списки є те, що вони не дозволяють змінювати розмір блоків пам'яті оптоелектронної системи (розбивати на менші або поєднувати) [7]. Тобто якщо програма запитуватиме тільки блоки одного розміру, система із вичерпуванням їхнього запасу розширюватиме динамічну пам'ять.

Методи близнят також мають ряд недоліків. За продуктивністю вони випереджають інші методи динамічного програмування, проте вони неефективні для малого об'єму даних через те, що пам'ять розбивають на блоки, розмір яких є степенем числа 2, що при невеликому об'ємі даних призводить до значної внутрішньої фрагментації, наслідок якої – неефективне використання обчислювальних ресурсів оптоелектронної системи [8].

Проаналізувавши наведені вище методи розподілення динамічної пам`яті визначено, що основними компонентами, які впливають на ефективність їхньої реалізації є структури даних, в яких наведена інформація про стан та характеристики фізичної пам`яті. На основі цього запропоновано новий метод. Цей метод буде базуватися на сукупності двох структур даних:

– Структура склейки даних, що забезпечує додаткову інформацію, про блоки пам'яті оптоелектронної системи: розмір попереднього і початкового блоків, інформацію, чи вільний блок.

– Структура пошуку даних, яка зберігає інформацію про усі вільні блоки оптоелектронної системи, що забезпечує швидкий пошук потрібного блоку;

Реалізація структури склейки даних мовою С має наступний вигляд:

typedef

```
struct _block {
        int last_size; //Розмір попереднього блоку
        int size; //Розмір даного блоку
        bool is_allocated; //Використання даного блоку
} block;
```

Структура пошуку даних «free_block» має використовувати три поля. Перше поле – вкладена структура склейки «block». Друге та третє поле – вказівники на попередню та наступну структуру «free_block». Подібний тип структури використовується в методі розподілення динамічної пам`яті за принципом вибору найкращого блоку. Необхідність використання структури такого типу обумовлена двома причинами. Перша причина – необхідність зберігання всіх вільних блоків пам`яті в одному списку. Друга причина – застосування даної структури зменшує фрагментацію. [2] Зберігання структури «free_block» найбільш доцільно виконати на початку вільного блоку.

В термінах мови С дана структура виглядає так:

```
typedef
struct _free_block {
        struct _block block;
        struct _free_block *last, *next;
} free_block;
```

Але застосування такої структури має один недолік – значна обчислювальна складність її застосування в обчислювальних системах.

Усунення даного недоліку можна виконати через використання вірно спланованими обчислювальними процедурами, які будуть використовуватися для розподілення динамічної пам`яті. Дане ствердження буде підтверджено обчислювальними експериментами, результати яких наведені у кінці статті.

Запропонований метод розподілення динамічної пам`яті складається з трьох процедур. Ці процедури наступні:

- процедура пошуку і виділення блоку пам'яті (функція `malloc`)
- процедура повернення вільного блоку пам'яті системі (функція `free` )
- процедура склеювання декількох сусідніх вільних блоків в один.

Проаналізуємо необхідні операції, які мають бути застосовані для кожної процедури.

Процедура пошуку. Основа цієї процедури – визначення в масиві структур «free_block» покажчика на перший вільний блок фізичної пам'яті необхідного розміру. Визначення даного покажчика найбільш доцільно виконувати за алгоритмом «пошук найкращого блоку». Цей алгоритм базується на знаходженні у фізичній пам’яті оптоелектронної системи вільного блоку, розмір якого найближчий до необхідного розміру запитаної динамічної пам’яті. Для зменшення часу роботи процедури пошуку необхідно взяти наступний у списку блок, якщо він такого ж розміру. В тому випадку, якщо розмір знайденого блоку збігається з необхідним розміром, блок видаляється зі списку і позначається як зайнятий, шляхом встановлення біту «is_allocated». В тому випадку, якщо розмір знайденого блоку більше, то блок розбивається на 2 блоки, один із яких необхідного розміру. Біт «is_allocated» цього блоку встановлюється в одиницю, а сам відповідний блок фізичної пам’яті видаляється зі списку. Блок фізичної пам’яті, який залишився після розподілення вставляється у відповідну позицію списку структур вільних блоків. Графічне представлення роботи процедури пошуку, яка використовує алгоритм «пошук найкращого блоку» наведено на рисунку 1.

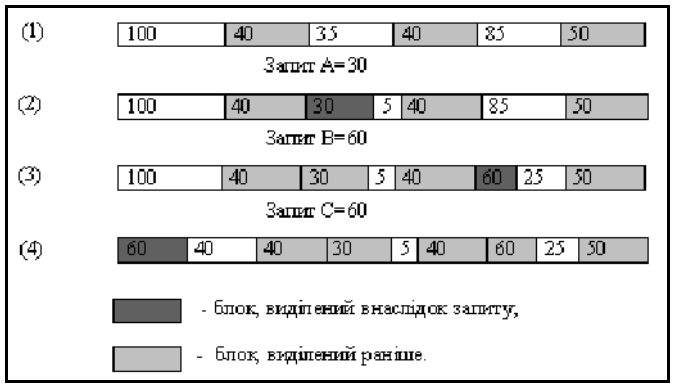


Рис. 1. Графічне представлення роботи процедури пошуку на основі алгоритму «пошук найкращого блоку»

Процедура повернення. Реалізація процедури повернення блоку фізичної пам'яті оптоелектронної системи виконано наступним чином. На першому етапі – біт «is_allocated», який міститься у відповідній структурі «block» для даного блоку фізичної пам'яті скидується. На другому етапі – відбувається вставка структури «free_block» у відповідну позицію списку.

Процедура склейки. Головний принцип роботи даної процедури полягає у поєднанні різних за розмірами вільних блоків фізичної пам'яті. Причина застосування даної процедури полягає у зменшенні фрагментації фізичної пам'яті [2]. Для роботи даної процедури необхідно застосовувати інформацію, яка наведена у полях структур «free_block» сусідніх вільних фізичних блоків. Якщо шляхом аналізу полів структури «free_block» один або обидва сусідніх блоків визначаються як вільні, то виконується склеювання – поєднання даних фізичних блоків пам'яті. На завершальному етапі процедури склеювання необхідно скорегувати поля структур «free_block» та виконати видалення структури «free_block» приєднаного фізичного блоку пам'яті зі списку структур блоків. Виконання в часі процедури склейки можна виконувати різним чином. Так, склеювання блоків може виконаються як відразу після звільнення блоку пам'яті оптоелектронної системи, так і через деякий проміжок часу. Одним з прикладів застосування другого варіанту ситуація, після якої нагромадилася певна кількість вільних блоків фізичної пам'яті. Операція склейки блоків є найбільш складним і дорогим, але принциповим з погляду мінімізації фрагментації.

Після проведення комп'ютерних експериментів по реалізації програмного варіанту запропонованого методу розподілення динамічної пам'яті в оптоелектронній системі отримані числові показники роботи – загальний час роботи запропонованого методу, час виділення блоку фізичної пам'яті, час звільнення блоку фізичної пам'яті та використаний об'єм пам'яті. Нижче наведено результати роботи створеного методу та кількох існуючих методів динамічного виділення пам'яті оптоелектронної системи, а саме: Doug Lea's malloc, Hans Boehm's garbage collector та Mike Haerthel's malloc.

Отримані результати наведені в таблиці 1. Ця таблиця також містить такі характеристики для найбільш поширених методів розподілення динамічної пам'яті.

Таблиця 1.

**Результати тестування різних методів реалізації динамічного виділення пам'яті**

| Реалізація | Загальний час, с | Час виділення блоку, с | Час повернення блоку, с | Використано пам'яті, байт |
|---|---|---|---|---|
| Doug Lea's malloc | 4,78 | $2{,}01*10^{-6}$ | $2{,}38*10^{-6}$ | 2752512 |
| Hans Boehm's garbage collector | 15,74 | – | – | 6967296 |
| Запропонований метод | 4,58 | $1{,}63*10^{-6}$ | $2{,}09*10^{-6}$ | 2770856 |
| Mike Haerthel's malloc | 4,21 | $0{,}58*10^{-6}$ | $2{,}01*10^{-6}$ | 3596288 |

## ВИСНОВКИ

Під час проведення наукового дослідження було запропоновано та програмно реалізовано метод розподілення динамічної пам'яті. Для застосування даного методу використовуються дві структури, в яких наведена інформація про характеристики та стан блоків фізичної пам'яті оптоелектронної системи. Програмний варіант реалізації даного методу складається з трьох процедур – пошуку, повернення та склеювання. Програмна реалізація запропонованого методу була порівняна на предмет швидкодії з іншими розповсюдженими методами розподілення динамічної пам'яті. Для оцінки швидкодії була написана програма, яка випадковим чином виділяла і повертала через певний час блоки розміром від 5 до 2000 байт. Оцінювався як і загальний час виконання програми, так і окремо середній час виконання операцій виділення та повернення блоків пам'яті оптоелектронної системи.

Аналізуючи дані таблиці 1, запропонований метод, який розташовано в третій стрічці, має значно кращі показники, в порівнянні з методом Hans Boehm's garbage collector: даний метод використовує на 15,74 % більше пам'яті та на 243,7% більше часу. Метод Mike Haerthel's malloc хоч і показує на 8,1% менші витрати часу, проте, використовує на 29,8% більше пам'яті, тому також значно програє, порівняно з новостворений методом, оскільки більша швидкодія досягається завдяки використанню надмірної кількості пам'яті оптоелектронної системи. Досить непогані показники має метод Doug Lea's malloc, але в порівнянні з новостворений, він хоч і показує на 0,7% менші затрати пам'яті, але затрати часу збільшуються на 4,4%.

З наведених даних видно, що існує співвідношення між швидкодією і використаною пам'ятю. Більш швидкий метод використовує більше пам'яті. Але на прикладі розробленого методу було продемонстровано, що підвищення швидкодії для виділення динамічної пам'яті блоками розміром від 5 до 2000 байт, не вимагає використання фізичної пам'яті оптоелектронної системи великого об'єму.

## СПИСОК ЛІТЕРАТУРИ


1. Christopher W. Fraser. A Retargetable C compiler: / Christopher W. Fraser. David R. Hanson. Design and Implementation. – Benjamin/Cummings Publishing, 1995.
2. A Memory Allocator. [Electronic resource] / Doug Lea // Hamser Verlag. – 1996. – Access mode: http://www.g.oswego.edu./dl/html/malloc.html.
3. Stephenson C. J. Fast fits: New method for dynamics storage allocation // Operating Systems Review 17(5), 1982, pp. 30-32.
4. J. L. Peterson langnp 1033, T. A. langnp1033 Norman. Langnp 1033 Buggy systems. Communications of the ACM, 20(6): 421-431, 1877.
5. G.O. Collins. Experience in automatic storage allocation. Communications of the ACM, 4(10): 436-440, October 1961.
6. Mark S. Johnstone, Paul R. Wilson. The Memory Fragmentation Problem: Solve / Mark S. Johnstone, Paul R. Wilson. – Texas, 2003, pp.36.
7. Donald E. Knuth. The Art of Computer Programming, volume 1: Fundamental Algorithms, Addison-Wesley, Reading, Massachusetts, 1973, pp.205.
8. Шехофцев В. А. Операційні системи – К.: Видавнича группа BHV, 2005. – 256 с.


Надійшла до редакції 20.12.2012 р.

**БЕВЗ О. М. – кандидат технічних наук, доцент, кафедра АІВТ, Вінницький національний технічний університет, Україна.**

**ВАСЮРА А. С. – академік УТА, професор каф. АІВТ, директор ІнАЕКСУ, Вінницький національний технічний університет, Україна.**

**ЛІСОВЕНКО А. І. – магістрантка групи 2КСУА-12м, кафедри АІВТ, Вінницький національний технічний університет, Україна.**