УДК 681.3

МОДИФІКАЦІЯ ІТЕРАЦІЙНОГО АЛГОРИТМУ КОМПОНОВКИ НА БАЗІ ГІПЕРГРАФОВОЇ МОДЕЛІ ЕЛЕКТРИЧНОЇ СХЕМИ

Білой О.І., Струнілін В.М.
Донецький національний технічний університет
кафедра комп'ютерної інженерії
E-mail: vstrun@ukr.net

Анотація

Білой О.І., Струнілін В.М. Модифікація ітераційного алгоритму компоновки на базі гіперграфової моделі електричної схеми. *Розглянуті алгоритми компоновки електричних схем в конструктивні елементи різних рівнів. Запропоновано модифікований ітераційний алгоритм компоновки, який більш ефективно, ніж стандартний метод, дозволяє виконувати компоновку при завданні електричної схеми у виді гіперграфу.*

Загальна постановка проблеми

Компоновка - одна з основних процедур, що виконуються при конструкторському проектуванні РЕА [1]. Це процес переходу від електричної схеми до конструктивного розподілу (розбиттю) всіх елементів на групи, у відповідності до конструктивів різних рівнів, тобто процес перетворення функціонального опису апаратури в конструктивне.

Завданням компоновки є розрізання великої схеми, якою може бути структурна, функціональна, логічна, електрична принципова, на частини.

Відомо декілька груп алгоритмів компоновки [1, 2]. Як правило, в алгоритмах компоновки математичною моделлю об'єкту є граф, вершини якого відповідають модулям, а ребра — між модульним з'єднанням.

У послідовних алгоритмах спочатку вибирається перша вершина графа і послідовним приєднанням до неї інших вершин з числа нерозподілених (за умови задоволення заданим критеріям) формується перший шматок графа. Потім набирається другий шматок графа і так далі до повного розрізання.

Завдання компоновки найчастіше вирішується змішаними алгоритмами в два етапи: початкова компоновка послідовними алгоритмами і поліпшення результатів початкової компоновки ітераційними процедурами до задоволення прийнятих критеріїв.

Також останнім часом популярності набули еволюційні алгоритми або алгоритми генетичного пошуку [1]. Вирішуючи оптимізаційні завдання за допомогою генетичних алгоритмів, розробники спираються на модель простого генетичного алгоритму, який далеко не завжди дає бажані результати.

Ітераційні алгоритми розрізання застосовуються для поліпшення початкової компоновки, отриманої в результаті виконання послідовних алгоритмів. У графі, розрізаному на підграфи проводиться парна перестановка елементів різних підграфів з перевіркою на кожному кроці приросту числа ребер, що сполучають куски. Мета ітерацій — мінімізація числа сполучних ребер.

В даному алгоритмі необхідно знайти пару вершин $x_i \in X^1, x_j \in E \setminus X^1$, для яких прирощення числа зовнішній зв'язків (ланцюгів) буде більше за 0 та максимальне.

$$\max \Delta S_{ij} = S_{ij} - S_{ji} > 0 \ ;$$

$$S_{ij} = | \Gamma X^1 \cap \Gamma E \setminus X^1 |, \ S_{ji} = | \Gamma X^1 \setminus X_i \cup X_j \cap \Gamma E \setminus X^1 \setminus X_j \cup X_i |,$$

де: X^1 - кусок матриці Q,

$E \setminus X^1$ - матриця Q без куска X^1,

S_{ij} –число зовнішніх зв'язків між двома кусками до перестановки вершин і та j,

S_{ji} –число зовнішніх зв'язків між кусками після перестановки вершин і та j.

$$\Delta S_{ij} = | \Gamma X^1 \cap \Gamma E \setminus X^1 | - | \Gamma X^1 \setminus X_i \cup X_j \cap \Gamma E \setminus X^1 \setminus X_j \cup X_i | .$$

Таким чином, матриця Q розбивається на 2 підматриці X^1 та $E \setminus X^1$ і для кожної пари вершин X_{ij} розраховується ΔS_{ij}. Потім серед усіх ΔS_{ij} шукаємо найбільше додатне число і міняємо місцями і-ту та j-ту строки матриці. Далі знову розраховуємо ΔS_{ij} для усіх можливих пар вершин підматриць. Якщо найбільше ΔS_{ij} дорівнює нулю або негативне та закінчуємо компоновку.

Цей алгоритм досить легко запрограмувати, але він має суттєвий недолік – у кожному циклі необхідно заново розраховувати ΔS_{ij} для усіх пар вершин підматриць, бо після того, як ми міняємо місцями строки матриці змінюється число зовнішніх та внутрішніх зв`язків кожного вузла схеми.

Ітераційний алгоритм компоновки гіперграфа на основі 4-х правил (Алгоритм 1)

Цей алгоритм на багато складніше запрограмувати ніж попередній і він потребує більше пам`яті у процесі роботи, але має суттєвий плюс – у кожному циклі необхідно розраховувати $\Delta S_{ij}$ тільки для тих пар вершин які пов`язані ребрами з переставленими на попередньому кроці.

Суть ітераційних алгоритмів полягає у виборі деякого початкового «розрізання» початкового гіперграфу схеми на куски з подальшою мінімізацією числа ланцюгів між кусками за допомогою парного обміну вершин різних кусків. При цьому для кожної ітерації здійснюється перестановка тих вершин, які забезпечують максимальне зменшення числа зв'язків (ланцюгів) між парою кусків.

У роботах [3, 4] сформульовані умови знаходження приросту числа зовнішніх зв'язків (ланцюгів) ΔS_{ij} при перестановці вершин $e_i \in G_n, e_j \in G_m$.

Для ланцюгу V_k, інцидентному вершині e_i, можливі наступні випадки:

1) ланцюг V_k буде видалений з розрізу при перестановці пари елементів e_i, e_j, якщо $E_k (E_k = \Gamma V_k)$ не містить e_j і жодній з вершин з множини $E_n \setminus e_i$, тобто

$$(E_k \setminus e_i) \bigcap (E_n \bigcup e_j) = O; \quad (1)$$

2) ланцюг V_k з'явиться в розрізі при перестановці пари елементів e_i, e_j, якщо E_k не містить жодної з вершин множини $E_м$, тобто

$$E_k \bigcap E_m = O. \quad (2)$$

Аналогічно для ланцюга V_l, інцидентній вершині $e_j \in E'_k$:

1) ланцюг V_l піде з розрізу при перестановці e_i, e_j, якщо $E_l (E_l = \Gamma V_l)$ не містить вершини e_i і жодній з вершин $E_m \setminus e_j$, тобто

$$(E_l \setminus e_j) \bigcap (E_m \bigcup e_i) = O; \quad (3)$$

2) ланцюг V_l з'явиться в розрізі при перестановці вершин e_i, e_j, якщо E_l не містить жодної з вершин множини E_l, тобто

$$E_l \bigcap E_n = O. \quad (4)$$

Якщо одночасно не виконуються умови (3) і (4), то ребро V_l залишиться в розрізі

після перестановки e_i, e_j. Очевидно, що якщо ребро залишиться в розрізі, то воно не впливає на приріст ΔS_{ij} і його можна не враховувати.

$$\Delta S_{ij} = (\bar{S}_i + \bar{S}_j) - (\overset{+}{S}_i + \overset{+}{S}_j), \quad (5)$$

де: $\bar{S}_i, \bar{S}_j$ - кількість ребер, які будуть видалені з розрізу;

$\overset{+}{S}_i, \overset{+}{S}_j$ - кількість ребер, які з'являться в розрізу при перестановці вершин e_i, e_j.

На кожній ітерації знаходимо таку пару вершин $e_i, e_j (e_i \in G_n \,\&\, e_j \in G_m)$, для якої $\max \Delta S_{ij} > 0$.

Після обміну пари e_i, e_j значення ΔS необхідно перерахувати для пар вершин, суміжних з елементами $e_i \cup e_j$. Алгоритм закінчується, коли $\Delta S_{ij} \leq 0$.

Модифікований ітераційний алгоритм компоновки гіперграфа (Алгоритм 2)

Матриця Q спочатку розбивається на підматриці $Q_0, Q_1, \cdots, Q_k$, що відповідає розбиттю гіперграфа на шматки $G_0, G_1, \cdots, G_k$. У підматриці Q_0 включаємо елемент e_0, який відповідає роз'єму схеми. Розгледимо дві підматриці Q_n і Q_m, які відповідають шматкам G_n і G_m гіперграфа G. Виходячи з (1)–(4) можна сформулювати наступні правила, стосовно матриці Q [5].

Правило 1. Ланцюг V_k залишається в розрізі при перестановці вершин $e_i, e_j \in E[e_i \in E_n \,\&\, e_j \in E_m]$ якщо:

а) у стовпці V_k матриці Q є два одиничні елементи в рядках e_i, e_j. У стовпці V_l матриці Q є два одиничні елементи в строках e_i, e_j.

б) у стовпці V_k матриці Q знайдеться два одиничні елементи (за винятком рядка e_i), один з яких належить підматриці Q_n, а другий — Q_m. У стовпці V_l матриці Q знайдеться два одиничні елементи (за винятком рядка e_j), один з яких належить підматриці Q_m, а другий— Q_n.

Правило 2. Ланцюг V_k піде з розрізу при перестановці вершин e_i, e_j, якщо в стовпці V_k матриці Q один одиничний елемент знаходиться в рядку e_i підматриці Q_n, а усі інші одиничні елементи знаходяться в підматриці Q_m. Ланцюг V_l піде з розрізу при перестановці вершин e_i, e_j якщо в стовпці V_l матриці Q один одиничний елемент знаходиться в рядку e_j підматриці Q_m, а усі інші одиничні елементи знаходяться в підматриці Q_n.

Правило 3. Ланцюг V_k з'явиться в розрізі при перестановці вершин $e_i \in E_n, e_j \in E_m$, якщо всі одиничні елементи знаходяться в підматриці Q_n. Ланцюг V_l з'явиться в розрізі, якщо всі одиничні елементи знаходяться в підматриці Q_m.

Слід зазначити, що при виконанні правил 1(б), 2 і 3 в стовпці $V_k (V_l)$ є один одиничний елемент в рядку $e_i (e_j)$.

Відмітимо, що при перестановці вершин e_i, e_j приріст ΔS_{ij} знаходиться лише для шматків G_n та G_m і, відповідно, для підматриці Q_n та Q_m.

Ясно, що перевірка ланцюгів по правилах 1-3 значно простіше, чим по формулах (1) – (4) і потребує менше часу на реалізацію значень ΔS_{ij}.

Тестування алгоритмів

Тестування алгоритмів проводилось випадково згенерованим гіперграфом з кількістю ребер та вершин від 10 до 100. При цьому гіперграфи “розбивались” на 2 - 10, 15, 20, 30, 40 та 50 кусків з рівною кількістю вершин у кожному. Результати досліджень залежності часу обчислення декількох з алгоритмів від розміру матриці приведені у таблиці 1.

Таблиця 1 – Залежність часу обчислення алгоритмів від розміру матриці Q

№	Потужність вершин X	Потужність ребер E	Число вузлів	Час рішення, сек.	
				Алгоритм 1	Алгоритм 2
1	20	20	2	0,165	0,055
2	40	40	2	0,934	0,22
3	80	80	2	4,286	0,495
4	100	100	2	10,165	0,989
5	100	100	3	48,297	5,275
6	100	100	4	51,978	5,934
7	100	100	5	91,319	12,365
8	100	100	10	110,275	20,275
9	100	100	20	39,725	8,846
10	100	100	50	16,648	4,066

По результатах досліджень, наведених у табл. 1 можна зробити висновок, що швидкодія алгоритму 1 відносно алгоритму 2 не лінійна і залежить від багатьох факторів. Гістограма швидкості обчислення наведена на рис.1. Графік залежності часу обчислення графу від кількості вузлів у схемі при розбитті на 7 підсхем приведений на рис.2.

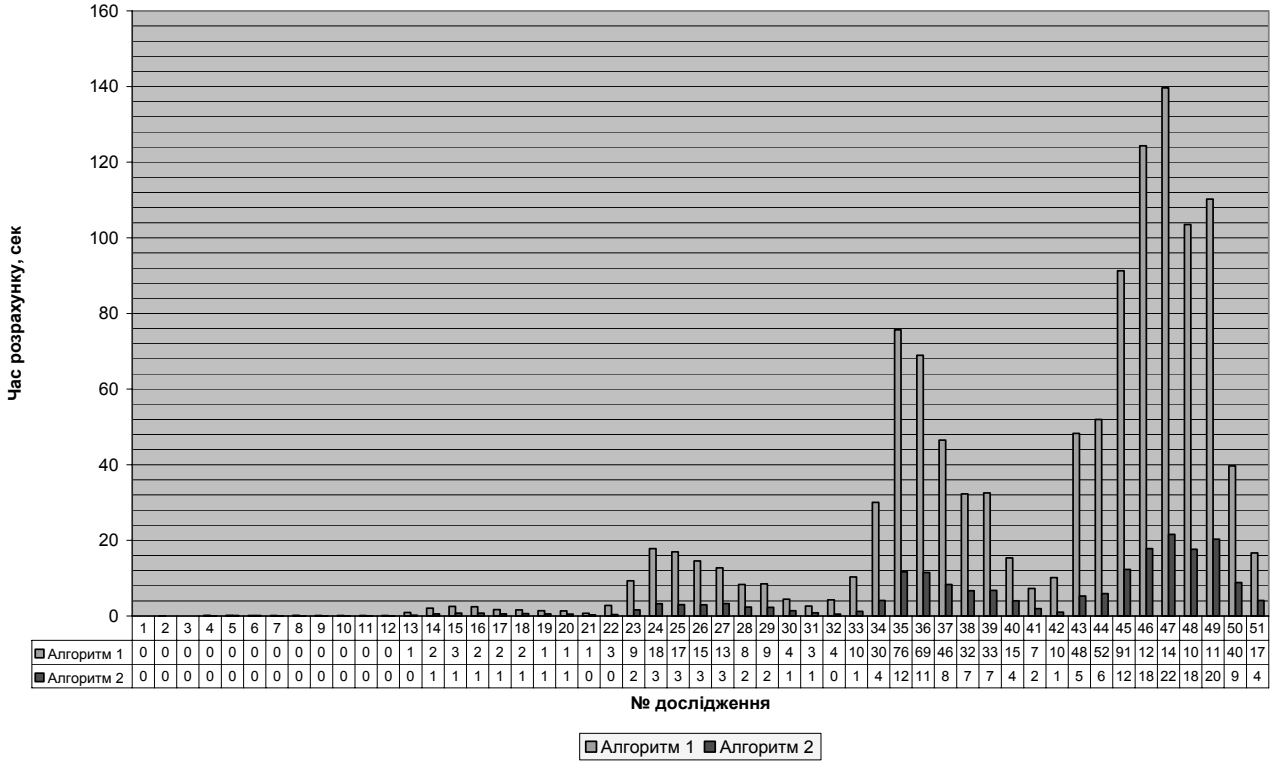

Рисунок 1 - Гістограма швидкості обчислення алгоритмів, що досліджуються

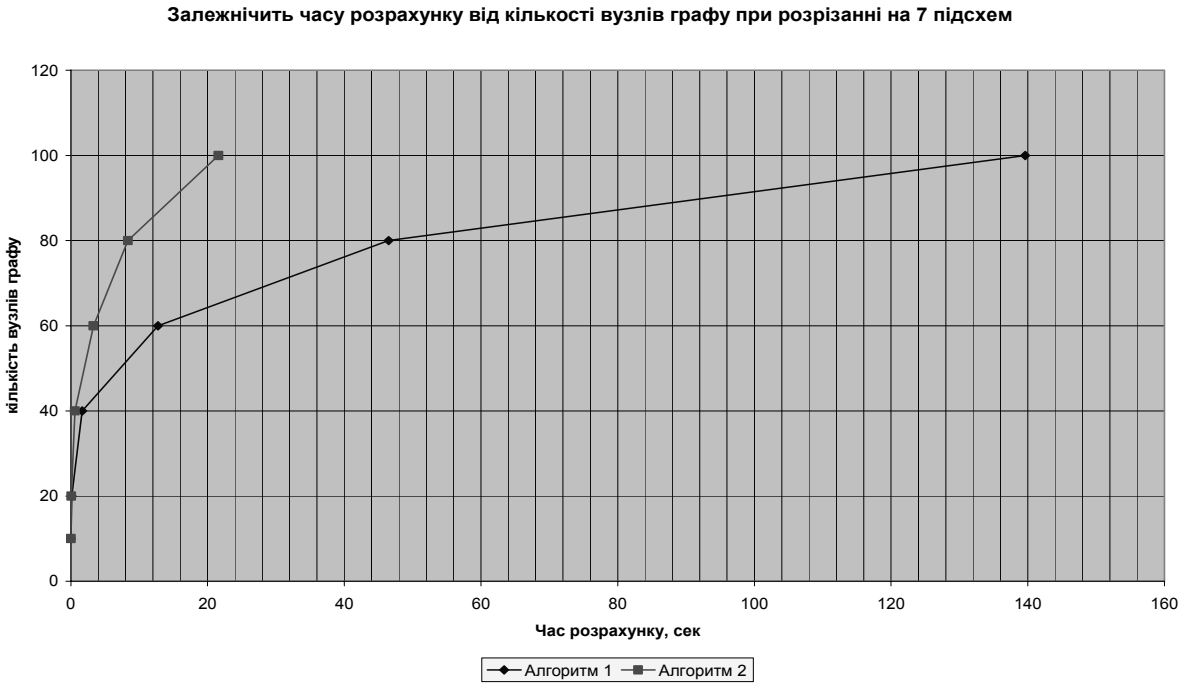

Рисунок 2 - Графік залежності часу розрахунку від розміру матриці при розрізанні на 7 підсхем

Висновки

Гіперграфова модель є найбільш зручною формою представлення електричних схем у пам'яті комп'ютера. За допомогою цієї системи був досліджений пропонований алгоритм компоновки.

По результатах дослідження виявилось, що пропонований алгоритм дозволяє компонувати схеми в 2-20 разів швидше (чим більша кількість вершин у графі, тим більший коефіцієнт ефективності пропонованого алгоритму відносно стандартного) ніж стандартний ітераційний алгоритм компоновки. І це на схемах, які містять до 100 вузлів. На реальних схемах, число вузлів у котрих може сягати сотень тисяч або навіть мільйонів, цей показник може значно збільшитись. Запропонований алгоритм не потребує більше пам'яті у процесі роботи ніж стандартний, а також не поступається результативністю та надійністю.

Список літератури

1. Курейчик, В. М. Математическое обеспечение конструкторского и технологического проектирования с применением САПР [Текст] / В. М. Курейчик. – М.: Радио и связь, 1990 . – 352 с.

2. Лебедев, Б. К. Методы поисковой адаптации в задачах автоматизированного проектирования СБИС: Монография / Б. К. Лебедев. – Таганрог: изд-во ТРТУ, 2000 . – 192 с.

3. Алексеев, О. В. Автоматизация проектирования радиоэлектронных средств: Учеб. пособие для радиотехн. спец. вузов [Текст] / Под ред. О.В. Алексеева. — М., 2000. — 479 с.

4. Мироненко, И. Г. Автоматизированное проектирование узлов и блоков РЭС средствами современных САПР: Учеб. пособие вузов / Под ред. И.Г. Мироненко. — М., 2002. — 391 с.

5. Саломатин, В.А. Итерационный алгоритм распределения конструктивных элементов при задании электрической схемы в виде гиперграфа [Текст] / В.А.Саломатин, В.Н.Струнилин // Наукові праці Донецького національного технічного університету. Серія «Інформатика, кібернетика і обчислювальна техніка» (ІКОТ-2008). Випуск 9 (132). — Донецьк: ДоНТУ. — 2008. — 316 с.