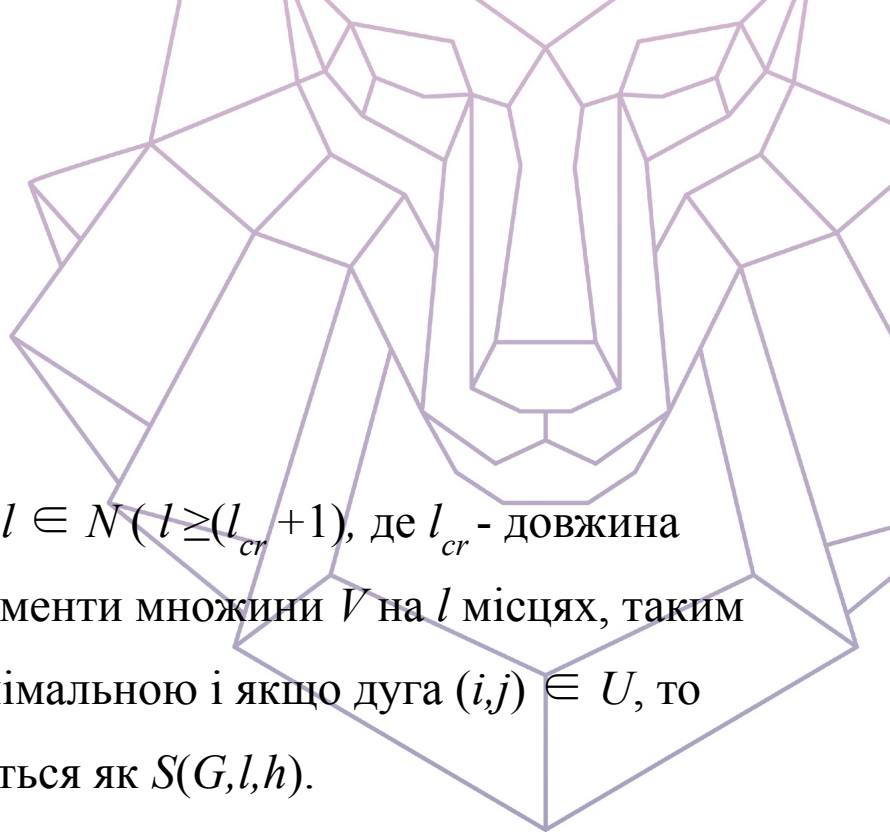
Дипломна робота на тему:
“Точні та наближені алгоритми мінімізації числа виконавців при заданих директивних термінах”

Виконавець: студентка групи ПС-17м-1 Джанашия Л.Р.
Керівник: доц. кафедри ОМ та МК Турчина В. А.

Практичні задачі мінімізації числа виконавців

- Задача планування розкладу руху поїздів, літаків, громадського транспорту.
- Планування зайнятості персоналу.
- Планування етапів розробки програмного забезпечення та розподіл завдань між розробниками.

Постановка задачі



Задано орграф $G = (V,U)$ і значення константи $l \in N$ ($l \geq (l_{cr} + 1)$, де l_{cr} - довжина критичного шляху). Необхідно розмістити елементи множини V на l місцях, таким чином, щоб ширина упорядкування h була мінімальною і якщо дуга $(i,j) \in U$, то елемент i стояв раніше j. Така задача позначається як $S(G,l,h)$.

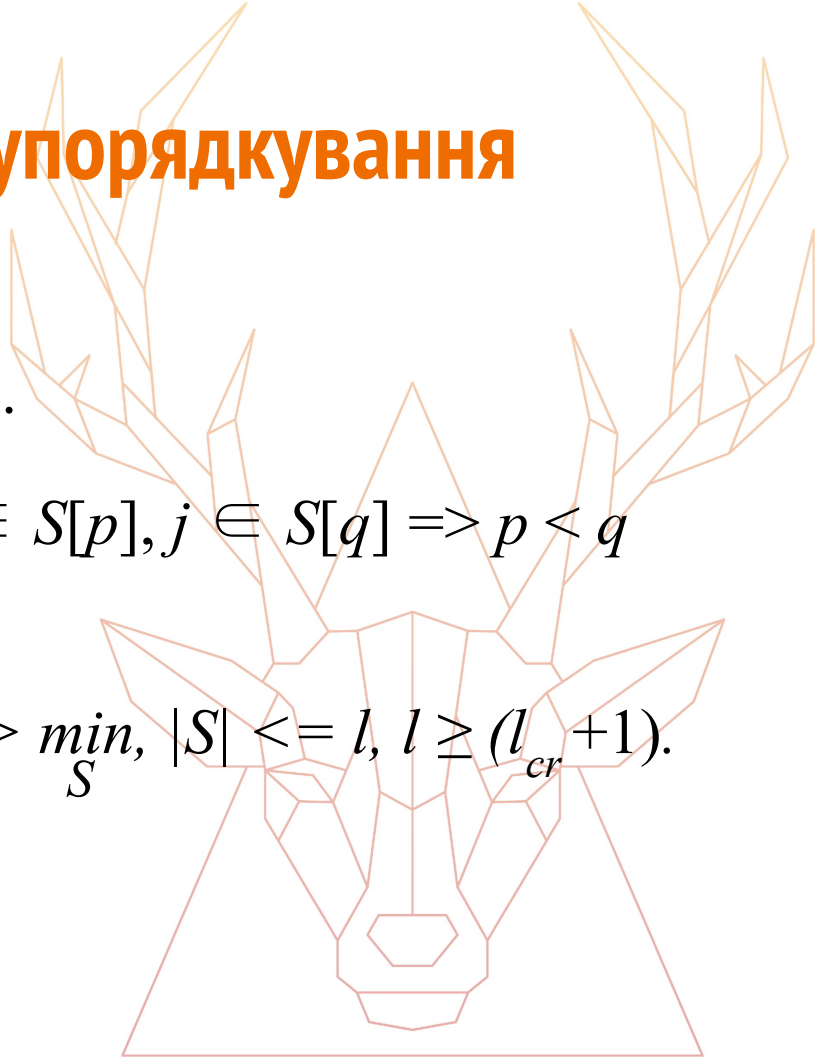
Оптимальне упорядкування

$G = \{V,U\}, |V| = n.$

$S: (i,j) \in U \ \& \ i \in S[p], j \in S[q] => p < q$

Задача:

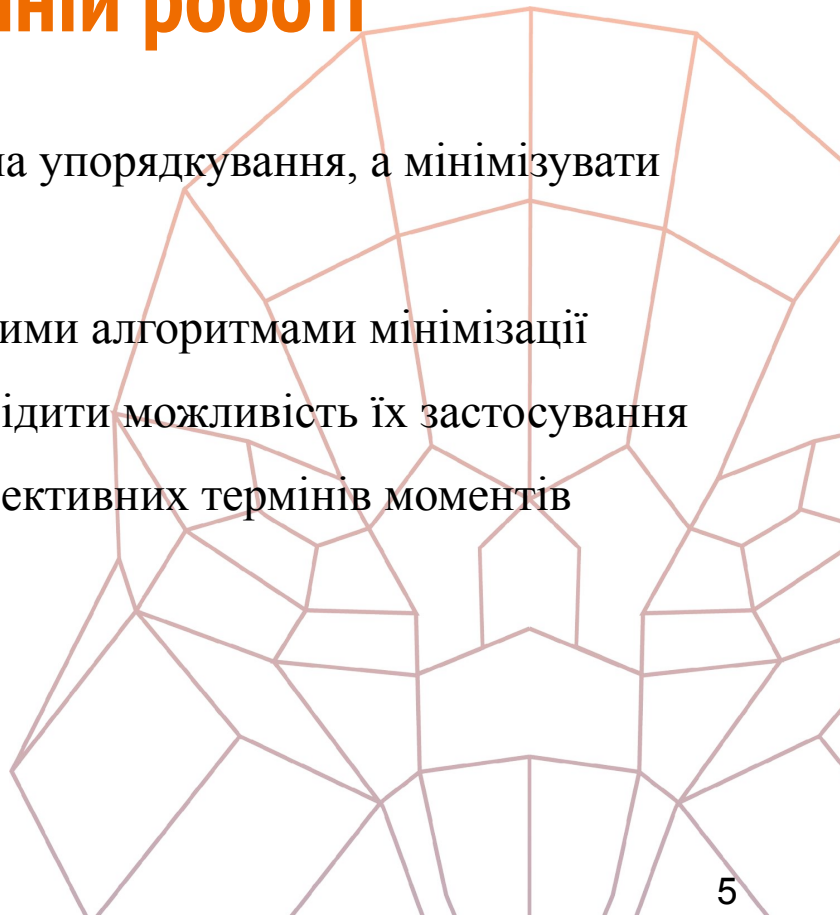
$h(S) = \max_{1<= i <= l} |S[i]| \text{ -> } \min_{S}, \ |S| <= l, \ l \geq (l_{cr}+1).$



Задача поставлена в дипломній роботі

Більш розвиненою є теорія, коли задана ширина упорядкування, а мінімізувати необхідно довжину.

Була поставлена задача ознайомитися з існуючими алгоритмами мінімізації ширини упорядкування заданої довжини та дослідити можливість їх застосування для розв'язку задачі для якої введені поняття директивних термінів моментів завершення робіт.



Алгоритми побудови спеціальних упорядкувань

Алгоритм побудови $\underline{S}$

$\underline{S} = S(G,n,l)$, $n = |V|$.

0. Вважаємо в $\underline{S}$ всі місця порожніми.

1. k:= 1.

2. В орграфі G шукаємо вершини, що не мають вхідних дуг, і записуємо їх на k-те місце в $\underline{S}$. Викреслюємо з G ці вершини й дуги, що виходять з них. Отриманий граф позначимо як G'.

3. Якщо множина вершин графа G' порожня, то – пункт 5.

4. k:= k+1, переходимо на пункт 2, вибираючи в якості орграфа G орграф G'.

5. Кінець.

Довжину отриманого упорядкування $\underline{S}$ позначимо як l_0.

Алгоритм побудови $\overline{S}$

0. Вважаємо в S всі місця порожніми.

1. $k := l_0$.

2. В орграфі G шукаємо вершини, що не мають вихідних дуг, і записуємо їх на k-те місце в S. Викреслюємо з G ці вершини й дуги, що входять в них. Отриманий граф позначимо як G'.

3. Якщо множина вершин графа G' порожня, то – пункт 5.

4. $k := k-1$, переходимо на пункт 2, вибираючи в якості орграфа G орграф G'.

5. Кінець.

Алгоритм знаходження інтервалів місць вершин

$l = |\overline{S}| = |\underline{S}|;$

$M = \{(g_v, r_v) \mid 1 \le g \le l,\ 1 \le r \le l,\ v \in V\},$

де g_v - ліва границя інтервалу місць вершини v, $v \in V$,

r_v – права границя інтервалу місць вершини v, $v \in V$.

$|M| = |V|$.

Алгоритм побудови множини M

1. $\forall\ i,\ i = \overline{(1, l)},\ \forall\ v,\ v \in \underline{S}[i],\ g_v = i.$

2. $\forall\ i,\ i = \overline{(1, l)},\ \forall\ v,\ v \in \overline{S}[i],\ r_v = i.$

3. Алгоритм завершив свою роботу.

Алгоритм побудови упорядкування $S(G, h, l)$ з використанням спеціальних упорядкувань

1. Будуємо $\underline{S}$ та $\overline{S}$.

1.1 Будуємо множину M інтервалів місць вершин. Фіксованими вершинами будемо називати ті, для яких $g_v - r_v = 0$, а нефіксованими відповідно для яких $g_v - r_v > 0$.

1.2 Записуємо упорядкування S_{fix}, що включає тільки фіксовані вершини, видаляючи з упорядкування S нефіксовані вершини.

$l' = \underline{l} + 1$, де $\underline{l}$ – довжина критичного шляху;

$S'_{fix} = S_{fix}$;

$|S'_{fix}| = l'$;

$h_{fix} = h(S_{fix})$;

l – задана довжина;

$h =]\frac{|V|}{l}[$.

1.3 Побудуємо матрицю I розміру n x n, де $n = |V|$, V – множина вершин.

$I_{ij} = 1$, якщо існує шлях з вершини v_i у вершину v_j,

$I_{ij} = 0$ в іншому випадку.

2. Побудова S'_{fix} для заданого l. Введемо список *Offset*. $|Offset| = l'$. $Offset' = Offset$. Значення елементів в *Offset* по замовчуванню дорівнюють 0.

2.1 Якщо $l' < l$, то $\forall\ i,\ i = \overline{(1, l')}$, якщо $|S'_{fix}[i]| > h$, то зміщуємо $|S'_{fix}[i]| - h$ вершин на місце i+1, а всі вершини починаючи з позиції i+1 також відповідно зміщуємо на 1 позицію вправо. $l' = l' + 1$. $|Offset'| = |Offset'| + 1$. $Offset'_i = 1$;

$|S'_{fix}| = |S'_{fix}| + 1$.

Для зміщення обираємо такі ($|S'_{fix}[i]| - h$) вершин, для яких значення $\sum_{j=1}^{n} I_{kj}$, k – номер вершини в $S'_{fix}[i]$, є мінімальним. Тобто обираємо вершини з найменшою кількістю шляхів до інших вершин.

$Offset = Offset'$.

2.2 Обчислюємо заново ліві та праві границі інтервалів місць для кожної нефіксованої вершини.

$\forall\ j, j = \overline{(1, n)}$:

$$r'_j = r'_j + \sum_{k=1}^{r_j} Offset_k,$$

$$g'_j = g_j + \sum_{k=1}^{g_j} Offset_k,$$

де r_j та g_j права та ліва границі інтервалів місць для вершини v_j відповідно.

Причому значення $Offset_k$ приймаємо за 0, якщо для кожної вершини v_i, що належить множині $S'_{fix}[g_j]$, $i = \overline{(1,n)}$, значення $I_{ij} = 0$. $r_j = r'_j$, $g_j = g'_j$.

3. Розміщення нефіксованих вершин.

3.1 Сортуємо нефіксовані вершини в порядку неспадання довжини інтервалів місць вершин.

3.2 Всередині кожної множини унікального діапазону сортуємо вершини в порядку неспадання лівої границі інтервалу.

Для кожної вершини у відсортованому списку, починаючи з лівої границі інтервалу, шукаємо вільне місце в S'_{fix}. Можливі 3 випадки:

а) Якщо для вершини vi обрали не крайню ліву позицію, а зі значенням offset > 0, то всі ліві границі вершин, які більше лівої границі вершини v_i та до яких v_i має шлях, зміщуємо на величину offset вправо.

б) Якщо для вершини v_i у вказаному діапазоні не знайшлося вільного місця та $l' = l$, то $h = \max(h, h_{fix})$ і переходимо на пункт 3.3.

Якщо для вершини v_i у вказаному діапазоні не знайшлося вільного місця та $l' < l$, то розміщуємо цю вершину на місці $r_i + 1$ (тобто після правої границі), а всі заповненні місця починаючи з $(r_i + 1)$-го зміщуємо на 1 вправо. Заново обчислюємо ліві та праві границі інтервалів:

$\forall$ v_j у відсортованому списку, якщо $r_j \geq r_i$ $(i \neq j)$ та $I_{ij} = 1$, то $r_j = r_j + 1$.

$\forall$ v_j у відсортованому списку, якщо $g_j \geq g_i$ $(i \neq j)$ та $I_{ij} = 1$, то $g_j = g_j + 1$.

$|Offset| = |Offset| + 1$

Зміщуємо значення $Offset_t$, $t = \overline{((r_i + 1), (|Offset|))}$ на 1 позицію вправо. $Offset_{ri+1} = 0$;

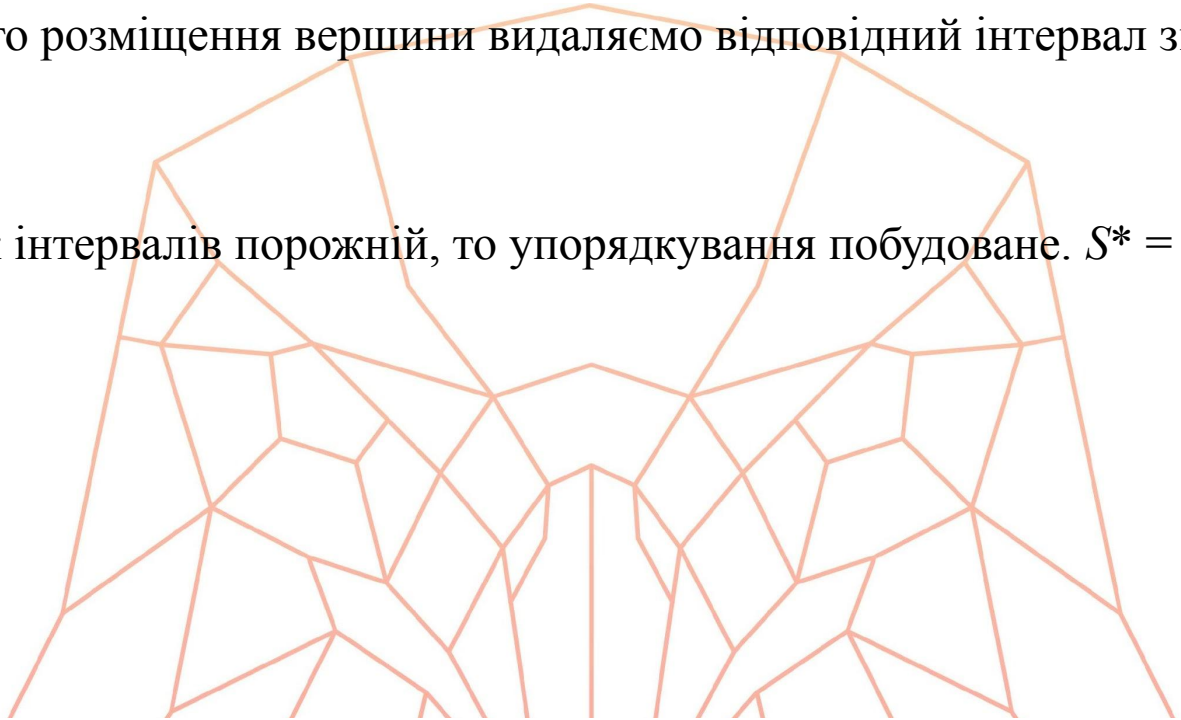
$l' = l' + 1$;

$|S'_{fix}| = |S'_{fix}| + 1$.

в) Якщо для вершини v_i знайшлося вільне місце, що дорівнює крайній лівій границі діапазону, то просто розміщуємо її в упорядкуванні.

Після кожного розміщення вершини видаляємо відповідний інтервал зі списку інтервалів.

Якщо список інтервалів порожній, то упорядкування побудоване. $S^* = S'_{fix}$.



Приклад роботи алгоритму

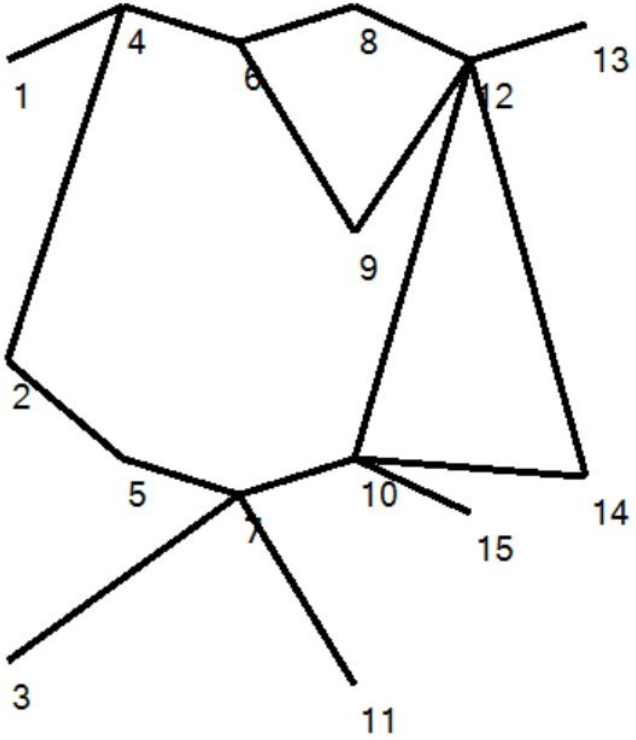
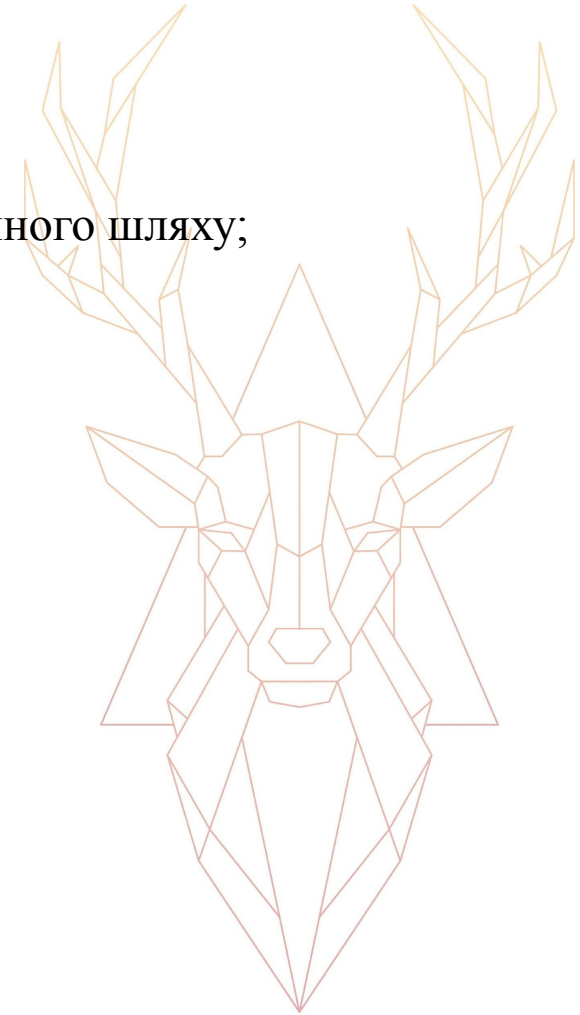


Рисунок 1 - Орієнтований граф
(орієнтованість зліва направо)

1. $\underline{S} = \langle 1\ 2\ 3 \mid 4\ 5 \mid 6\ 7 \mid 8\ 9\ 10\ 11 \mid 12\ 15 \mid 14\ 13 \rangle$ $\bar{S} = \langle 1\ 2 \mid 4\ 5\ 3 \mid 6\ 7 \mid 8\ 9\ 10 \mid 12 \mid 14\ 13\ 15\ 11 \rangle$

1.1 Інтервали місць нефіксованих вершин: $M_3 = [1, 2]$, $M_{11} = [4, 6]$, $M_{15} = [5, 6]$.

1.2 $S_{fix} = \langle 1\ 2 \mid 4\ 5 \mid 6\ 7 \mid 8\ 9\ 10 \mid 12 \mid 14\ 13 \rangle$



$l' = l = 6$, l – довжина критичного шляху;

$S'_{fix} = S_{fix}$;

$|S'_{fix}| = 6$;

$h_{fix} = h(S_{fix}) = 3$;

$h = \left]\frac{|V|}{l}\right[= 2.$

1.2 Матриця $I =$

$$\begin{matrix}
0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{matrix}$$

|Offset| = 6.

l'= 6 < l = 8, отже ми можемо змістити одну з вершин на місці 4.

$\sum_{j=1}^{15} I_{8j} = 3;$

$\sum_{j=1}^{15} I_{9j} = 3;$

$\sum_{j=1}^{15} I_{10j} = 4.$

Отже серед вершин на місці $S'_{fix}[4]$ найвигідніше буде зміщати або вершину 8, або 9.

порядкування з фіксованих вершин має такий вигляд:

$$S'_{fix} = \begin{pmatrix} 1 & 2 \\ 4 & 5 \\ 6 & 7 \\ 8 & 10 \\ 9 & \\ 12 & \\ 14 & 13 \end{pmatrix};$$

: {0,0,0,1,0,0,0}, |Offset|= $|S'_{fix}|$ = l' = 7.

ирина всіх місць в упорядкуванні відповідає h,

на наступний крок.

За формулами зміщення інтервалів вершин у поточному пункті, ми маємо змістити інтервали M_{11} та M_{15}, оскільки при підраховуванні сум

$$r_{11} = 4 + \sum_{k=1}^{4} Offset_k,$$
$$g_{11} = 6 + \sum_{k=1}^{6} Offset_k,$$
$$r_{15} = 5 + \sum_{k=1}^{5} Offset_k,$$
$$g_{15} = 6 + \sum_{k=1}^{6} Offset_k,$$

елемент $Offset_4 = 1$.

Але враховуючи наступне зауваження у цьому пункті, ми бачимо, що на 5-му місці в упорядкуванні стоїть вершина 9, від якої немає шляху до 15-ї вершини, а на 4-му вершини 8 та 10 й від них також немає шляху до вершини 11. Отже $Offset_4$ ми не враховуємо й інтервали не зміщуємо.

3. Розміщуємо нефіксовані вершини.

3.1 $M_3 = [1, 2]$, $M_{15} = [5, 6]$, $M_{11} = [4, 6]$.

3.2 Інтервали є вже відсортваними за лівою границею в рамках кожної з довжин інтервалів.

3.3 Вершина 3 відповідає випадку б), оскільки для неї не знайшлося вільного місця в рамках інтервалу. Зміщуємо вправо місця починаючи з 3-го, а дану вершину розміщуємо на місці 3.

$$S'_{fix} = \left\langle \begin{matrix} 1\ 2 \\ 4\ 5 \\ 3 \\ 6\ 7 \\ 8\ 10 \\ 9 \\ 12 \\ 14\ 13 \end{matrix} \right\rangle$$

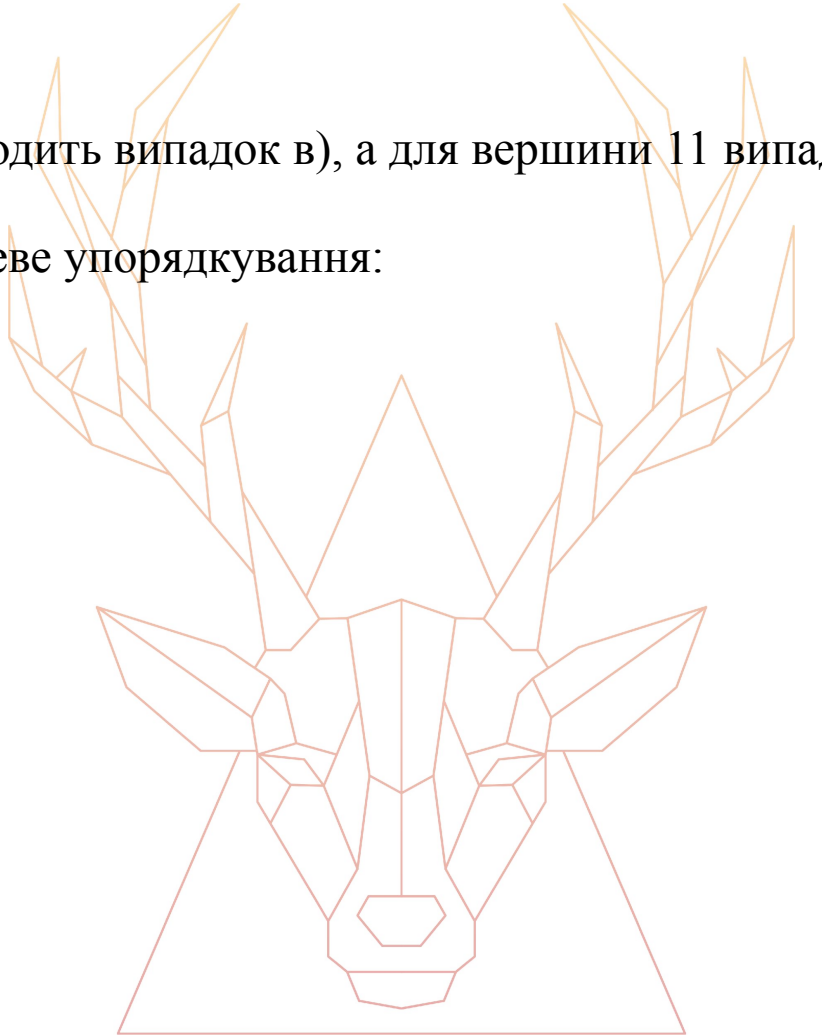
, 7],

, 7].

Для вершини 15 підходить випадок в), а для вершини 11 випадок а).

Отримаємо таке кінцеве упорядкування:

$$S^* = \left\langle \begin{matrix} 1\ 2 \\ 4\ 5 \\ 3 \\ 6\ 7 \\ 8\ 10 \\ 9\ 15 \\ 12\ 11 \\ 14\ 13 \end{matrix} \right\rangle.$$



Програмна реалізація

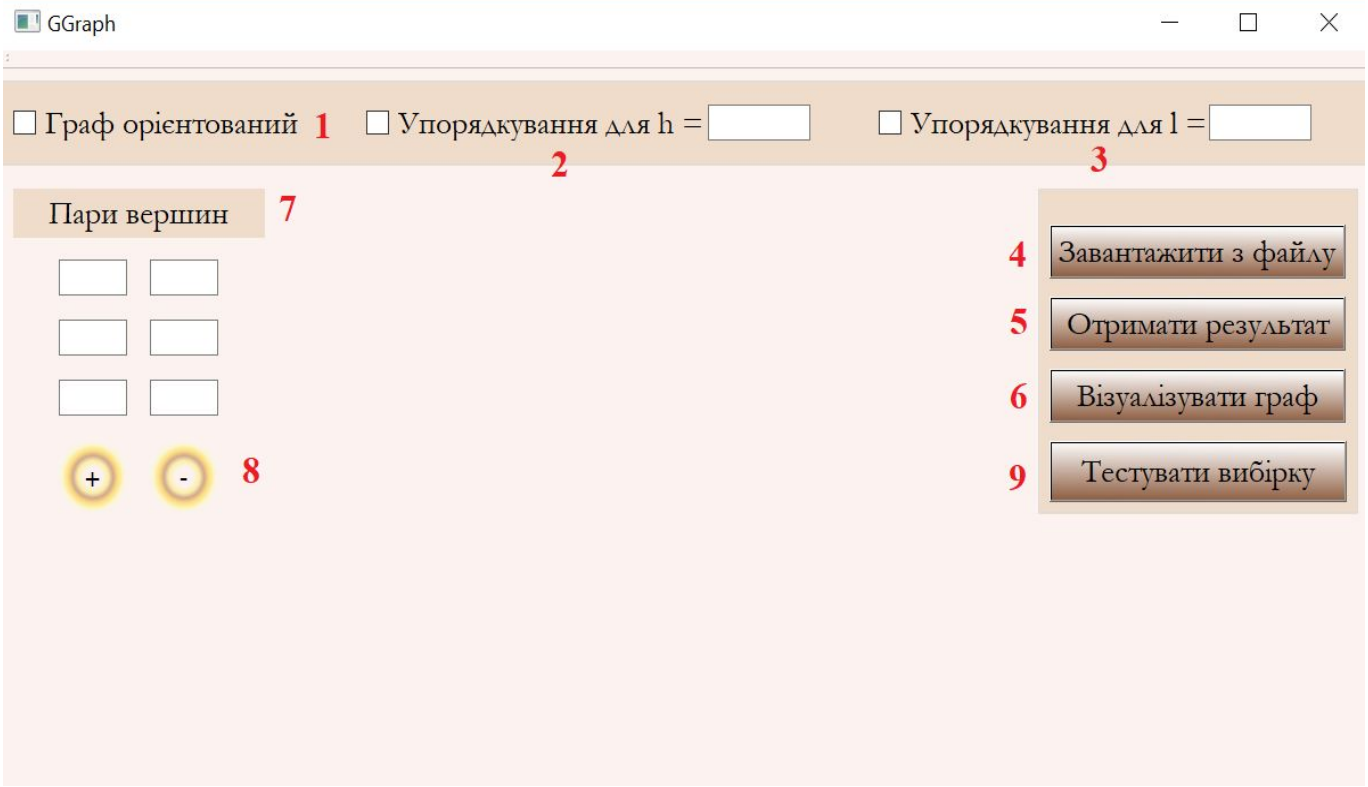


Рисунок 2 - Інтерфейс програми

Опис інтерфейсу

1 – Область для задання орієнтованості графу.

2 – Опція для побудови упорядкування заданної ширини h

3 – Опція для побудови упорядкування заданної довжини l

4 – Кнопка введення даних з файлу

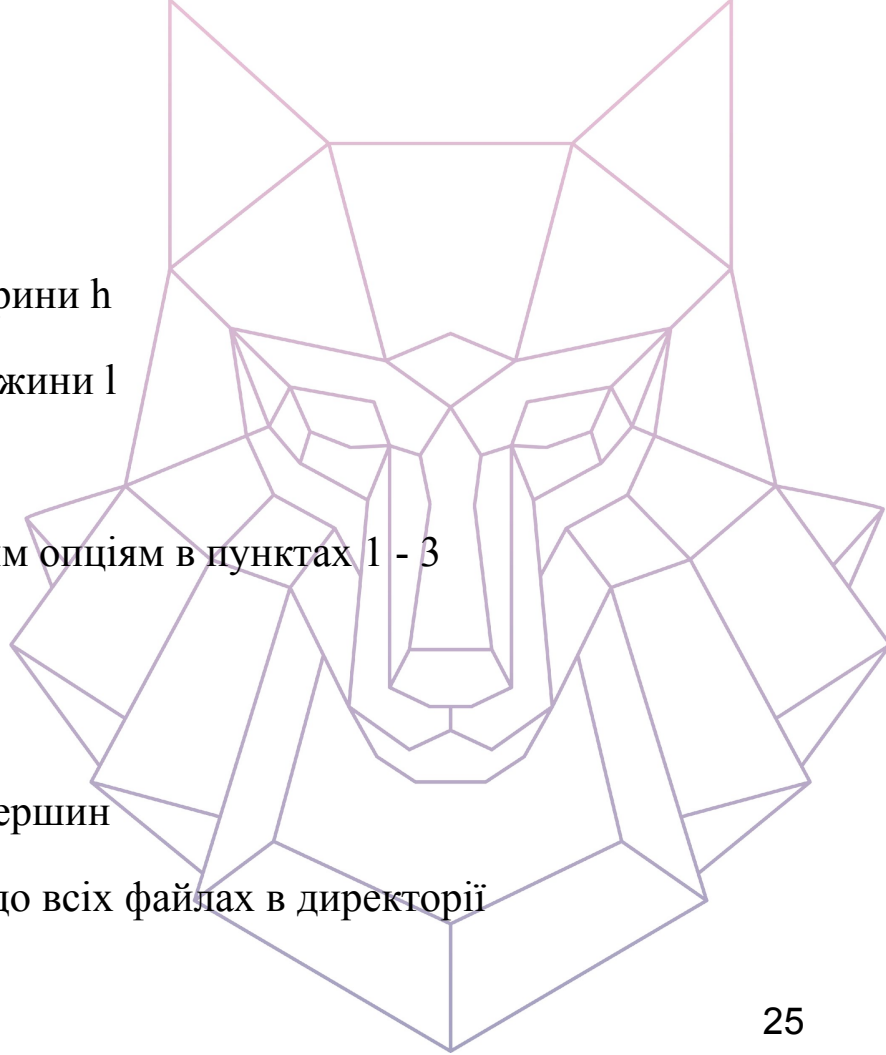
5 – Кнопка отримання результату відповідно обраним опціям в пунктах 1 - 3

6 – Кнопка візуалізації графу

7 – Поля для введення пар вершин вручну

8 – Кнопки для додавання (+) та видалення (-) пар вершин

9 – Кнопка, що дозволяє застосувати обрану опцію до всіх файлах в директорії



Тестування програми

- Вибірка - 118 ациклічних орієнтованих графів.

- Застосування алгоритму

  Для кожного ографу з вибірки будується упорядкування для 3-х довжин:

  - $l = l_{cr} + 1$, де l_{cr} дорівнює довжині критичного шляху;
  - $l = l_{cr} + 2$;
  - $l = l_{cr} + 3$.

  Отримані упорядкування зображені на рисунку 3.

Автоматично побудовані упорядкування

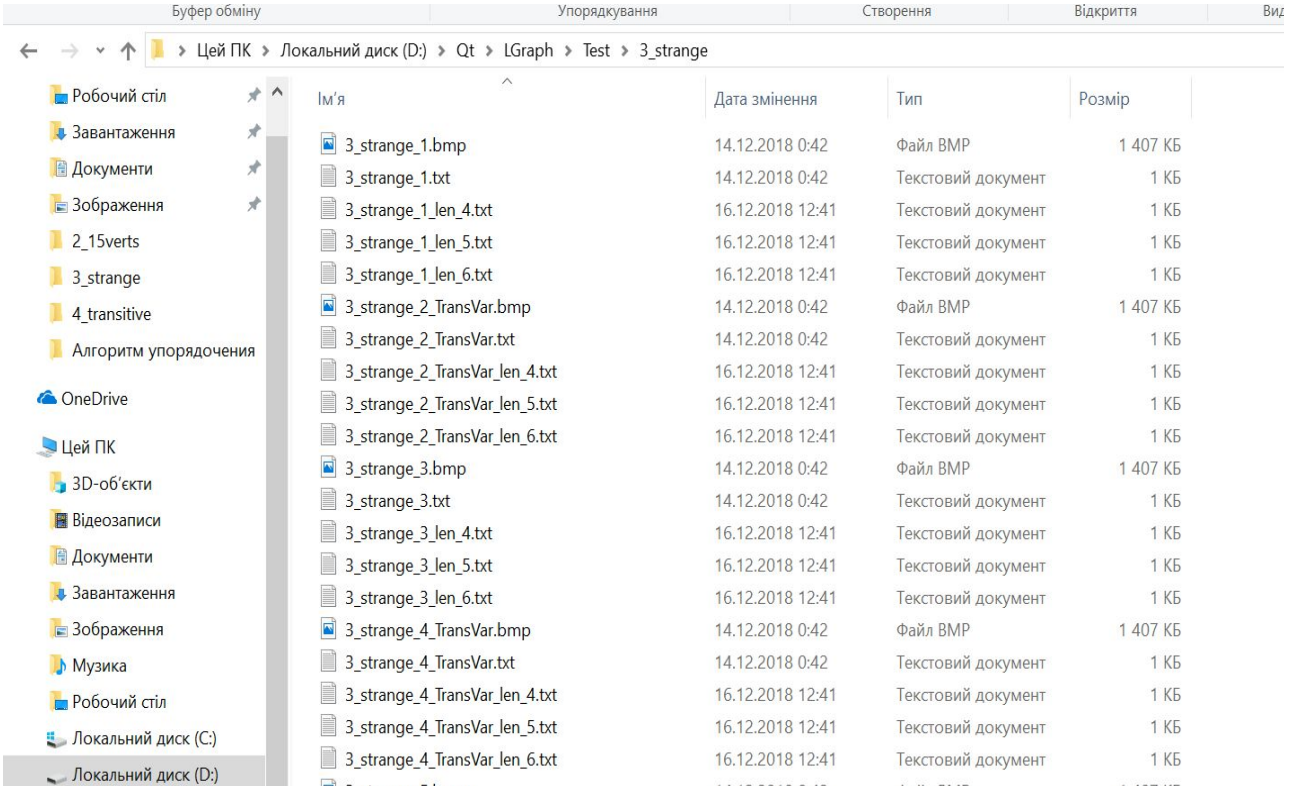


Рисунок 3 - Автоматично побудовані упорядкування

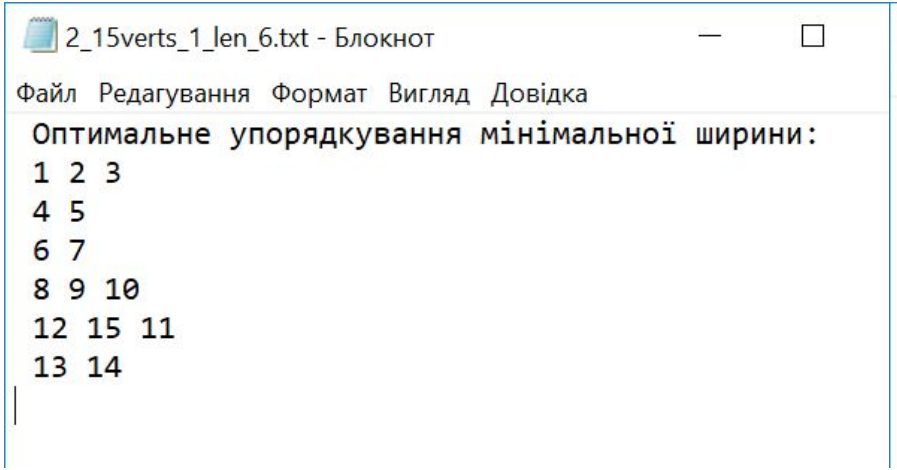


Рисунок 4 - Автоматично побудоване упорядкування довжини 6

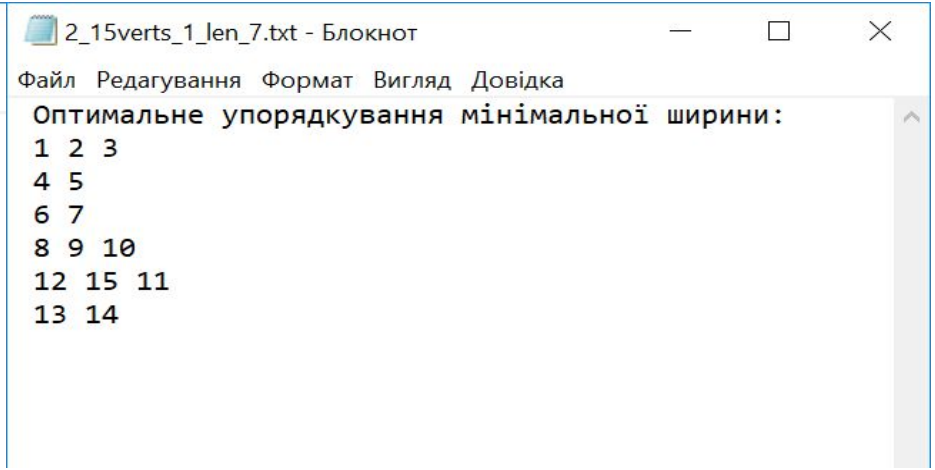


Рисунок 5 - Автоматично побудоване упорядкування довжини 7

Аналіз результатів тестування

Нехай l – задана довжина упорядкування;

S – побудоване упорядкування;

h(S) – ширина побудованого упорядкування;

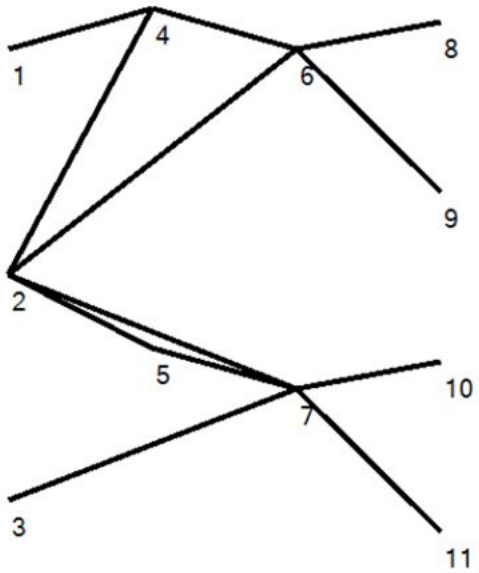
$h^* = \left]\frac{|V|}{l}\right[$, де V – множина вершин орграфу для якого побудоване упорядкування.

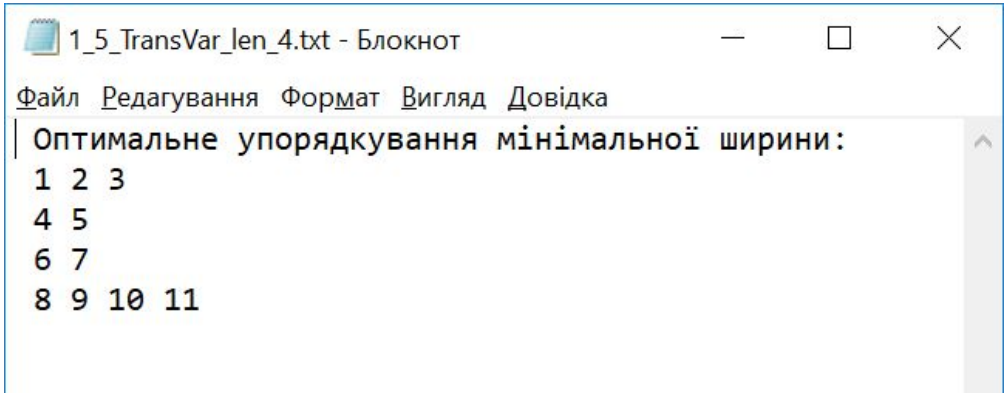
Якщо h(S) > h*, то помічаємо упорядкування як потенційно неоптимальне та зберігаємо результат у спеціально відведеній директорії.

Якщо в такому потенційно неоптимальному упорядкуванні можна мінімізувати ширину, то вважаємо, що алгоритм спрацював не задовільно,

якщо ж розташування вершин і дуг графу не дозволяє це зробити, вважаємо, що алгоритм спрацював задовільно.

Приклад потенційно неоптимального упорядкування





' - Неоптимальне упорядкування

Рисунок 6 - орграф, для якого
неможливо побудувати оптимальне упорядкування

Серед результатів вибірки не було виявлено орграфів, для яких було побудоване неоптимальне упорядкування саме через неточність алгоритму.

Висновки

- В роботі вивчений один із класів задач дискретної оптимізації, а саме, задача паралельного упорядкування.
- Розроблено алгоритм побудови мінімізації ширини упорядкування вершин орієнтованого ациклічного графа заданної довжини з врахуванням директивних інтервалів можливих місць вершин.
- Алгоритм програмно реалізований.
- Алгоритм простестовано та може вважатися задовільним по точності наближеним алгоритмом.

Дякую за увагу!