УДК 519.854.2

ЗАДАЧА ОПТИМАЛЬНОГО ПЛАНУВАННЯ РОБІТ ЗА НАЯВНОСТІ РІЗНОЇ ПРОДУКТИВНОСТІ ВИКОНАВЦІВ

Галкіна Г. А.

Національний технічний університет України “Київський політехнічний інститут імені Ігоря Сікорського”, Україна, Київ

доктор технічних наук, старший науковий співробітник, Гуляницький Л. Ф.

Інститут кібернетики імені В. М. Глушкова Національної академії наук України, Україна, Київ

Розглядається задача планування роботи на ітерацію гнучкої методології Scrum. Вона розглядається як задача оптимального планування робіт за наявності різної продуктивності виконавців. Характеристиками розглянутої задачі є наявність загального директивного строку для всіх пристроїв, різна непропорційна продуктивність пристроїв, наявність відношення часткового строгого впорядкування між роботами та визначена для кожного з завдань важливість. Наведено змістовну та математичну постановки задачі, що розглядається, та запропоновано евристичний алгоритм її розв’язування шляхом розбиття на дві підзадачі та розв’язування їх окремо. Для кожної з отриманих підзадач наведено математичні постановки. Для однієї з підзадач запропоновано жадібний алгоритм для отримання припустимого розв’язку.

Ключові слова: Scrum, оптимальне планування, розклад, різна продуктивність, директивний термін, паралельні пристрої, жадібний алгоритм.

Галкина Г. А, доктор технических наук, старший научный сотрудник, Гуляницкий Л. Ф. Задача оптимального планирования работ при наличии разной продуктивности исполнителей / Национальный технический университет Украины «Киевский политехнический институт имени Игоря Сикорского», Институт кибернетики имени В. М. Глушкова Национальной академии наук Украины, Украина, Киев

Рассматривается задача планирования работы на итерацию гибкой методологии Scrum. Она рассматривается как задача оптимального планирования работ при наличии разной продуктивности исполнителей. Характеристиками рассмотренной задачи являются наличие общего директивного срока для всех устройств, разная непропорциональная продуктивность устройств, наличие отношения частичной строгой упорядоченности между работами и определённая для каждого из заданий важность. Приведены смысловая и математическая постановки и предложен эвристический алгоритм решения рассматриваемой задачи путём разбития её на две подзадачи. Для каждой из полученных подзадач приведены математические постановки. Для одной из полученных подзадач предложен жадный алгоритм для получения допустимого решения.

Ключевые слова: Scrum, оптимальное планирование, расписание, разная продуктивность, директивный срок, параллельные устройства, жадный алгоритм.

H. Halkina, Dr.Sc. (Technology), Senior Researcher, L. Hulianytskyi The optimal scheduling problem for machines with different productivity / National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, V. M. Glushkov Institute of Cybernetics of National Academy of Sciences of Ukraine, Ukraine, Kyiv

The problem of scheduling work for an iteration in Scrum methodology is considered. It is considered as an optimal scheduling problem for machines with different unrelated productivity. The main characteristics of the problem include the existence of a common due date for all the machines, different unrelated productivity of the machines, the partial strict ordering set for the tasks and the value set for each of the tasks to be scheduled. The semantic and formal models of the problem are specified. A heuristic algorithm to solve the considered problem is suggested. It is based on the idea of splitting the problem into two different subproblems. For each of the resulting subproblems a formal model is specified. A greedy algorithm for finding a feasible solution is suggested for one of the subproblems.

Keywords: Scrum, optimal scheduling, schedule, different productivity, due date, parallel machines, greedy algorithm.

Вступ. Ітеративні методології розробки програмного забезпечення стають дедалі більш розповсюдженими у сучасному світі. Усі такі методології базуються на підході, який включає в себе виконання обмеженого обсягу завдань для проекту за певний час, що називається ітерацією.

Розглянемо ітеративну методологію, що має назву Scrum. Scrum – це підхід, в рамках якого можливо вирішити складні адаптивні проблеми, і в той же час продуктивно та із застосуванням творчого підходу розробити продукт найвищої якості. Scrum складається зі Скрам Команд (Scrum Teams), в яких розподілено відповідні ролі (roles), а також церемоній (events), артефактів (artifacts) та правил (rules) [1]. Серцем Scrum є Спринт, з часовими рамками в місяць або менше, в результаті якого створюється “завершений”, цінний та потенційно готовий до випуску Інкремент продукту. Тривалість

Спринту є постійною протягом усього періоду розробки. Кожен Спринт може вважатися проектом із часовими рамками в межах одного місяця. Як і інші проекти, Спринт використовується для досягнення певних цілей. Кожен Спринт складається із визначення того, що потрібно розробити, дизайну та гнучкого плану, які є орієнтирами при розробці, роботи та власне продукту, що стане результатом цієї роботи. [1]

Робота, що її виконують під час Спринту, планується під час наради із Планування Спринту. План дій розробляється при спільній роботі цілої Скрам Команди. Для Спринту тривалістю в місяць часові рамки зустрічі становлять вісім годин. Для більш коротких Спринтів на планування виділяють менше часу, пропорційно загальній довжині Спринту. Приміром, для двотижневого Спринту планування займе не більше чотирьох годин [1].

Тобто для кожного Спринту необхідно проводити планування, на якому має бути присутньою вся команда. Це планування займає серйозний, хоча й обмежений, проміжок часу. Саме тому автоматизація процесу планування роботи для одного Спринту принесе значну економію часу команди, яка працює над проектом. Таким чином, формалізація задачі планування роботи на ітерацію є актуальною з практичної точки зору.

Аналіз останніх досліджень і публікацій. Зазначена проблема є різновидом задач планування за умови різної продуктивності пристроїв. З наукової точки зору вона являє собою комбінацію задачі пакування декількох рюкзаків (Multiple Knapsack Problem [3]) та задачі планування роботи на паралельних пристроях з різною продуктивністю (Scheduling on Unrelated Parallel Machines [4]).

Задачу пакування декількох рюкзаків можна сформулювати наступним чином: маємо набір речей та набір рюкзаків. Кожна річ має

важливість та вагу, кожен рюкзак – місткість. Необхідно спакувати наявні рюкзаки таким чином, щоб максимізувати сумарну важливість речей, які до них потрапили. [3] Спостерігаємо явну аналогію між рюкзаками та робочим часом кожного учасника команди, також кожна річ має вагу (аналогія з часом, який витрачається на виконання завдання) та важливість.

Задачу планування роботи на паралельних пристроях з різною продуктивністю з наявним директивним терміном сформульовано наступним чином: маємо декілька пристроїв та набір завдань, які необхідно виконати на даних пристроях. Спільні елементи із задачею, що розглядається – складання розкладу для пристроїв із різною продуктивністю та наявність директивного терміну. Подібні задачі докладно розглянуто у [4].

Проблема планування Спринту у методології Scrum як математична задача є відносно новою та малодослідженою. Один із методів її розв'язання наведено у [5]. В даній публікації було запропоновано математичну постановку задачі з урахуванням ризиків та жадібний алгоритм для автоматичного планування Спринту. Інший погляд на проблему планування Спринтів наведено у [6]. Відмінністю від мети даної статті є те, що у [6] сформульовано задачу розподілу всіх завдань проекту по спринтах з метою максимізації важливості та мінімізації ризиків. Проте недоліком такого підходу є те, що за принципами гнучких методологій, до яких належить і Scrum [7], ми не маємо можливості планувати розподіл всіх робіт наперед, оскільки обсяг роботи з самого початку нефіксований і може змінюватися. Відповідно, доведеться проводити перепланування із новими вхідними даними.

Мета та завдання статті. Метою статті є формалізація та дослідження одного класу задач оптимального планування робіт за

наявності різної продуктивності пристроїв. Для досягнення даної мети необхідно виконати наступні завдання:

– сформулювати змістовну та математичну постановки для досліджуваної задачі оптимального планування робіт за наявності різної продуктивності пристроїв;

– запропонувати алгоритм розв'язування сформульованої задачі;

– сформулювати жадібний алгоритм для знаходження припустимого розв'язку для однієї з отриманих підзадач.

Постановка задачі. Для автоматизації процесу планування роботи команди на протязі обмеженого проміжку часу необхідно формалізувати дану задачу. Наведемо її змістовну постановку.

Для зручності введемо позначення $N = \{1,..,n\}$.

Нехай є команда, яка складається з m учасників.

Ця команда працює над проектом, для якого задано n завдань. Кожне з цих завдань може виконуватися будь-яким учасником команди, але тільки одним.

Також завдання є частково упорядкованими, тобто є такі завдання $j, j \in N$, для яких задане деяке завдання $k, k \in N \setminus \{j\}$, яке має бути виконане до них. Кожне завдання може передувати або одному завданню, або жодному.

Для кожного учасника команди заданий час виконання кожного з завдань даним учасником $t_{ij}, i = \overline{1,m}, j = \overline{1,n}$.

Для кожного завдання визначена його важливість $v_j, j = \overline{1,n}$.

Серед цих завдань є q обов'язкових до виконання завдань $Q = \{r, r \in N\}, |Q| = q$. Інші завдання не обов'язкові до виконання. Для визначеності без втрати загальності вважатимемо, що в списку завдань дані обов'язкові завдання є першими q завданнями. Також

зазначимо, що обов'язкові завдання не є частково впорядкованими, тобто для них не визначено такі завдання k, що їм передують або мають бути виконані після їх закінчення.

Заданий також директивний термін виконання підмножини завдань d.

Необхідно вибрати для виконання учасниками команди такий набір завдань, що обов'язково включає всі завдання з Q та щоб сумарна важливість цього набору завдань була максимальною, та скласти розклад виконання цих задач учасниками команди.

Математична модель

Задача полягає у максимізації цільової функції:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} v_j, \tag{1}$$

яка задає сумарну важливість усіх завдань, що будуть виконані, де

$$x_{ij} = \begin{cases} 1, \textit{якщо учасник } i \textit{ виконує завдання } j \\ 0, \textit{інакше} \end{cases}, i = \overline{1,m}, j = \overline{1,n}. \tag{2}$$

Також позначимо s_j час початку виконання завдання $j, j \in N$, причому:

$$s_j = \begin{cases} 0, \textit{якщо завдання } j \textit{ не буде виконане}, \\ > 0, \textit{якщо завдання } j \textit{ буде виконане}. \end{cases} \tag{3}$$

Обмеження задачі:

$$0 \le \sum_{i=1}^{m} x_{ij} \le 1, \tag{4}$$

$$\sum_{i=1}^{m} x_{ij} = 1, j \in Q. \tag{5}$$

Для всіх $j \in N, k \in N$ таких, що $x_{ij} = x_{ik}$ для всіх $i = \overline{1,m}$ та $\sum_{i=1}^{m} x_{ij} = 1, \sum_{i=1}^{m} x_{ik} = 1$:

$$s_j < s_k \to s_j + t_{ij} < s_k. \tag{6}$$

Для всіх завдань $j, j \in N$ таких, що $s_j > 0$:

$$s_j + \sum_{i=1}^{m} x_{ij} t_{ij} \le d. \quad (7)$$

Для всіх $j, j \in N \setminus Q$, для яких $\sum_{i=1}^{m} x_{ij} = 1$ та є завдання k, що їм передують:

$$s_k < s_j,\ s_k + \sum_{i=1}^{m} x_{ik} t_{ij} < s_j. \quad (8)$$

Обмеження (4) задає можливість виконання кожного з завдань лише одним учасником команди. (5) задає обов'язкове виконання всіх завдань $j, j \in Q$. (6) – неперетинання запланованих робіт для одного учасника. Обмеження (7) задає виконання всіх завдань в розкладі до настання директивного терміну, а (8) – виконання завдань відповідно до відношення часткового строгого впорядкування.

Запропонуємо наступний алгоритм розв'язування сформульованої задачі.

Розглянемо її як сукупність двох підзадач. Спочатку необхідно спланувати виконання обов'язкових завдань (підзадача 1), а після цього на час, що залишається, запланувати виконання такого набору необов'язкових завдань, який дасть максимально можливу важливість (підзадача 2).

Логічним буде в якості цільової функції для підзадачі 1 обрати мінімізацію сумарного часу, який буде витрачено на виконання обов'язкових завдань, оскільки наступним кроком є додавання необов'язкових завдань і на них також необхідний час.

Тоді на початок розв'язання підзадачі 2 отримаємо деякий існуючий розподіл обов'язкових завдань на всіх членів команди. З нього випливає, що для кожного члена команди вже є декілька

завдань, і це необхідно враховувати при складанні остаточного розкладу.

Наведемо математичні моделі підзадач 1 та 2.

Математична модель підзадачі 1 (планування обов'язкових завдань)

Задача полягає у мінімізації цільової функції

$$\sum_{i=1}^{m}\sum_{j=1}^{q} x_{ij}t_{ij}, \tag{9}$$

яка задає сумарний час, витрачений на виконання всіх обов'язкових завдань, де:

$$x_{ij} = \begin{cases} 1, \textit{якщо учасник } i \textit{ виконує завдання } j, \\ 0, \textit{інакше} \end{cases}, i = \overline{1,m},\ j = \overline{1,n}. \tag{10}$$

Обмеження задачі:

$$\sum_{i=1}^{m} x_{ij} = 1, j \in Q. \tag{11}$$

Для всіх учасників $i, j \in \{1,2,..,m\}$:

$$\sum_{j=1}^{q} x_{ij}t_{ij} \le d. \tag{12}$$

Обмеження (11) задає виконання кожного з обов'язкових завдань одним учасником команди. (12) задає виконання всіх обов'язкових завдань в розкладі кожного з учасників до настання директивного терміну.

Сформульована математична модель описує узагальнену задачу про призначення. Ця класична задача комбінаторної оптимізації є NP-складною [8], для неї вже було розроблено велику кількість різноманітних алгоритмів.

Сформулюємо математичну модель підзадачі 2. Для неї ми матимемо додаткові вхідні дані після розв'язування підзадачі 1 – розподілені між учасниками обов'язкові завдання з Q. Позначимо їх як $b_j, j \in Q$. Очевидно, що $b_j \in \{1,2,..,m\}$.

Математична модель підзадачі 2 (планування необов'язкових завдань)

Задача полягає у максимізації цільової функції

$$\sum_{i=1}^{m}\sum_{j=1}^{n} x_{ij} v_j, \tag{13}$$

яка задає сумарну важливість усіх завдань, що будуть виконані, де

$$x_{ij} = \begin{cases} 1, \textit{якщо учасник } i \textit{ виконує завдання } j, \\ 0, \textit{інакше} \end{cases}, i = \overline{1,m},\ j = \overline{1,n}. \tag{14}$$

Також позначимо s_j час початку виконання завдання $j, j \in N$, причому:

$$s_j = \begin{cases} 0, \textit{якщо завдання } j \textit{ не буде виконане}, \\ > 0, \textit{якщо завдання } j \textit{ буде виконане}. \end{cases} \tag{15}$$

Обмеження задачі:

$$0 \le \sum_{i=1}^{m} x_{ij} \le 1, \tag{16}$$

$$x_{b_j j} = 1, j \in Q. \tag{17}$$

Для всіх $j \in N, k \in N$ таких, що $x_{ij} = x_{ik}$ для всіх $i = \overline{1,m}$ та $\sum_{i=1}^{m} x_{ij} = 1, \sum_{i=1}^{m} x_{ik} = 1$:

$$s_j < s_k \to s_j + t_{ij} < s_k. \tag{18}$$

Для всіх завдань $j, j \in N$ таких, що $s_j > 0$:

$$s_j + \sum_{i=1}^{m} x_{ij} t_{ij} \le d. \tag{19}$$

Для всіх $j, j \in N \setminus Q$, для яких $\sum_{i=1}^{m} x_{ij} = 1$ та є завдання k, що їм передують:

$$s_k < s_j,\ s_k + \sum_{i=1}^{m} x_{ik} t_{ij} < s_j. \quad (20)$$

Обмеження (16) задає можливість виконання кожного з завдань лише одним учасником команди. (17) задає виконання обов'язкових завдань саме тими учасниками, яких було визначено в результаті розв'язання підзадачі 1. (18) – неперетинання запланованих робіт для одного учасника, (19) задає виконання всіх завдань в розкладі до настання директивного терміну. Обмеження (20) задає виконання завдань відповідно до відношення часткового строгого впорядкування.

Алгоритм знаходження припустимого розв'язку підзадачі 2. Пропонується жадібний алгоритм для знаходження припустимого розв'язку сформульованої підзадачі 2. Перевагами жадібних алгоритмів є невеликий час роботи та простота реалізації, отже даний клас алгоритмів є зручним для початку дослідження нових задач та їх розв'язування перед розробкою складніших алгоритмів.

Але для нашої задачі послідовність локально оптимальних (жадібних) виборів не дає в результаті глобального оптимального розв'язку [9]. Тому наведений далі алгоритм може бути використаний для знаходження деякого припустимого розв'язку, який не обов'язково буде оптимальним.

Для початку наведемо алгоритм сортування списку завдань таким чином, що у відсортованому списку завдань номер кожного завдання $k_j \in N, j \in N$ є більшим, ніж номер завдання, яке йому передує, тобто $r_j > r_k$. Такий алгоритм є модифікацією алгоритму сортування

включенням [9], в якому замість порівняння значень елементів, що сортуються, перевіряється відношення передування.

Для зручності позначимо $P = N \setminus Q, |P| = p, p = n - q$.

Алгоритм сортування завдань у порядку відношення передування

Для кожного завдання $j \in P$:

ЯКЩО в поточному списку перед даним завданням є завдання, які мають виконуватись після нього

ТО помістити поточне завдання на позицію в списку перед тими завданнями, яким воно передує

Запропонований алгоритм буде використаний у наведеному далі жадібному алгоритмі для того, щоб задовольнити обмеження на передування завдань.

Жадібний алгоритм для задачі про планування роботи команди на ітерації

Крок 1. Помістити всі обов'язкові завдання в початок розкладу відповідного учасника, впорядкувавши їх за порядковим номером.

Крок 2. Впорядкувати всі завдання $j, j = \overline{1, p}$ за алгоритмом сортування завдань у порядку відношення передування.

Крок 3. Для кожного завдання $j, j = \overline{1, p}$ із впорядкованої на кроці 2 послідовності:

3.1 ЯКЩО для поточного завдання j не визначено завдання k, яке йому передує,

ТО для всіх учасників команди $i = \overline{1, m}$:

ЯКЩО в розкладі даного учасника ще немає завдань та $t_{ij} < d$ АБО час завершення поточного завдання j, якщо його розмістити після останнього завдання в розкладі даного учасника $k \in N$ виконується $s_k + t_{ik} + t_{ij} < d$,

ТО додати поточне завдання до розкладу поточного учасника i та перейти до наступного завдання $j+1$.

ІНАКШЕ перейти до наступного учасника

3.2 ІНАКШЕ ЯКЩО завдання k вже у розкладі одного з учасників,

ТО для всіх учасників команди $i = \overline{1,m}$:

ЯКЩО в розкладі даного учасника ще немає завдань та $s_k + \sum_{i=1}^{m} x_{ik} t_{ik} + t_{ij} < d$

ТО додати поточне завдання до розкладу поточного учасника на час $s_j = s_k + \sum_{i=1}^{m} x_{ik} t_{ik}$;

АБО ЯКЩО час завершення поточного завдання j, якщо його розмістити після останнього завдання в розкладі даного учасника $r \in N$ $s_r + t_r + t_{ij} < d$ та $s_k + \sum_{i=1}^{m} x_{ik} t_{ik} < s_r + t_{ir}$,

ТО додати поточне завдання до розкладу поточного учасника на час $s_j = s_r + t_{ir}$

перейти до наступного завдання $j+1$

ІНАКШЕ перейти до наступного учасника

3.3 ІНАКШЕ перейти до наступного завдання $j+1$

В результаті роботи даного алгоритму буде отримано деякий припустимий розв’язок сформульованої підзадачі 2. Оскільки алгоритм є дуже примітивним, то отриманий розв’язок буде з високою імовірністю далеким від оптимального, але він надає можливість знайти деяку відправну точку для подальшого покращення

отриманого розв'язку за допомогою складніших евристичних та метаевристичних алгоритмів.

Висновки. В статті сформульовано змістовну та математичну постановки однієї задачі планування за умов різної продуктивності пристроїв. Запропоновано алгоритм розв'язування даної задачі за допомогою розбиття її на дві окремих підзадачі. Сформульовано математичні постановки підзадач із запропонованого методу розв'язання. Запропоновано жадібний алгоритм для розв'язання однієї з двох підзадач запропонованого алгоритму. Наведений матеріал може бути використаний для подальшого розв'язання сформульованої задачі планування за умов різної продуктивності пристроїв.

Література:

1. Посібник зі Скраму [Електронний ресурс] / К. Швабер, Д. Сазерленд – Режим доступу до ресурсу: https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-UA.pdf

2. Rubin K. EssentialScrum / KennethRubin. – Boston : Addison-Wesley, 2013. – 504 p.

3. Martello S. Knapsack Problems. Algorithms and Computer Implementations/ S. Martello, P. Toth. – New Jersey: John Wiley & Sons, 1990. – 296 p.

4. Brucker P. Scheduling Algorithms / P. Brucker. – New York : Springer Publishing, 2007. – 371 p.

5. Jansi S. A Greedy Heuristic Approach for Sprint Planning in Agile Software Planning / S. Jansi, K. C. Rajeswari // International Journal for Trends in Engineering & Technology. – 2015, vol.3. – №1. – P.18-21.

6. Golfarelli M. Sprint Planning Optimization in Agile Data Warehouse Design/ M. Golfarelli, S. Rizzi, E. Turricchia // 14th International

Conference, DaWaK 2012, Vienna, Austria, September 3-6, 2012. Proceedings. – P.30-41.

7. Agile. Гнучкі методології [Електронний ресурс] – Режим доступу до ресурсу: http://moyaosvita.com.ua/menedzhment/agile-gnuchki-metodologii/

8. The Generalized Assignment Problem and Its Generalizations [Електронний ресурс] / M. Yagiura, T. Ibaraki – Режим доступу до ресурсу: http://leeds-faculty.colorado.edu/glover/TS%20-%20vignettes%20-%20GAP%20Yagiura%20&%20Ibaraki.pdf

9. Кормен Т. Алгоритмы. Построение и анализ / Т. Кормен, Ч.И. Лейзерсон, Р.Л. Ривест, К. Штайн. – М.: МНЦМО, 2002. – 960 с.

References:

1.The Scrum Guide [Elektronnyj resurs] / K.Schwaber, J.Sutherland – Rezhym dostupu do resursu: https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-UA.pdf

2. Rubin K. Essential Scrum / Kenneth Rubin. – Boston: Addison-Wesley, 2013. – 504 p.

3. Martello S. Knapsack Problems. Algorithms and Computer Implementations / S. Martello, P. Toth. – New Jersey : John Wiley & Sons, 1990. – 296 p.

4. Brucker P. Scheduling Algorithms / P. Brucker. – New York: Springer Publishing, 2007. – 371 p.

5. Jansi S. A Greedy Heuristic Approach for Sprint Planning in Agile Software Planning / S. Jansi, K. C. Rajeswari // International Journal for Trends in Engineering & Technology. – 2015, vol.3. – №1. – P.18-21.

6. Golfarelli M. Sprint Planning Optimization in Agile Data Warehouse Design/ M. Golfarelli, S. Rizzi, E. Turricchia // 14th International Conference, DaWaK 2012, Vienna, Austria, September 3-6, 2012. Proceedings. – P.30-41.

7. Agile. Ghnuchki metodologhiji [Elektronnyj resurs] – Rezhym dostupu do resursu:http://moyaosvita.com.ua/menedzhment/agile-gnuchki-metodologiï/

8. The Generalized Assignment Problem and Its Generalizations [Elektronnyj resurs] / M. Yagiura, T. Ibaraki – Rezhym dostupu do resursu: http://leeds-faculty.colorado.edu/glover/TS%20-%20vignettes%20-%20GAP%20Yagiura%20&%20Ibaraki.pdf

9. Cormen T. Introduction to Algorithms / T. Cormen, C. Leiserson, R. Rivest, C. Stein. – London: The MIT Press, 2009. – 960 c.