УДК 628.52.012.011.56

МОНИТОРИНГ ЗАДАЧ В СИСТЕМАХ КООРДИНАЦИИ ВЗАИМОДЕЙСТВУЮЩИХ АГЕНТОВ

М.П. РЕВОТЮК, О.В. КОТ
Белорусский государственный университет информатики и радиоэлектроники,
П. Бровки, 6, Минск, 220013, Республика Беларусь
Поступила в редакцию
Аннотация. Рассматривается процесс регулярного решения классической линейной асимметричной задачи о назначении, когда множества работ и исполнителей, а также локальные оценки назначения формируются в реальном времени. Предложена структура данных и инкрементальный алгоритм построения оптимального паросочетания, в которых накопление предопределенных решений снижает вычислительную сложность решения до линейной зависимости от объема поступивших данных.

Ключевые слова: метод кратчайшего пополняющего пути, динамическая задача о назначении.

Abstract. The process of regular solution of the classical linear asymmetric assignment problem is considered, when sets of works and performers, as well as local assignment estimates are formed in real time. A data structure and an incremental algorithm for constructing optimal matching are proposed, in which the allocation of predefined solutions reduces the computational complexity of the solution to linear dependence on the volume of incoming data.
Keywords: shortest augmenting path method, dynamic assignment problems.

Введение

Задачи оптимизации управления системами взаимодействующих агентов естественно формулируются в терминах задач о динамическом назначении [1-3]. В процессе координации таких систем необходимо регулярно решать задачу о назначении агентам возникающих задач с учетом реальных ограничений и возможной коррекции плана назначения. Традиционно задачи координации агентов сводятся к известным задачам дискретной оптимизации, таким как задача о назначении или задача нескольких странствующих коммивояжеров. Однако необходимость учета реальных отношений между агентами и задачами приводит к экспоненциальной сложности алгоритма формирования оптимального назначения [3,4]. Процесс управления системами взаимодействующих агентов преследует цель формирования оптимального отображения множества задач T на множество агентов A. Функция, определяющая дискретный процесс принятия решения по распределению агентов как одного из видов ресурсов для решения задач, часто записывается в рекуррентной форме связи в момент времени t переменных состояния S_t, вектора решения X_t, функции прогноза состояния V_t и оценками эффективности управления S_t. Например, для реализации метода динамического программирования [1] такая функция имеет вид:

$$X_t^* = \arg\max\{c_t x_t + E[V_{t+1}(S_{t+1}(x_t)) \mid S_t] \mid x_t \in X_t\}. \quad (1)$$

При принятии решения о назначениях нет полной информации о наличии ресурсов и задач в будущем, а также об эффективности или приемлемости назначения. Стандартный инженерный подход состоит в близоруком решении задачи о назначении, используя только информацию, доступную на момент принятия решения [1]. Отображение (паросочетание) $X: T \to A$ в практически интересных случаях определяется решением линейной задачи о назначении (ЛЗН), для которой оценки эффективности назначения заданы матрицей $C: T \times A \to R^+$ [2]. При этом предолагается, что множества T и A формируются независимо и асинхронно (рис. 1), хотя иногда зависимости между элементами таких множеств отражают введением фиктивных элементов.

Классические ЛЗН в матричной постановке имеют вид

$$\min\left\{\sum_{i=1}^{n}\sum_{j=1}^{n}c_{ij}\cdot x_{ij}\,\middle|\,\sum_{i=1}^{n}x_{ij}=\sum_{j=1}^{n}x_{ij}=1;\ x_{ij}\ge 0;\ i,j=\overline{1,n}\right\} \qquad (2)$$

Случай поиска максимума сводится к постановке вида (2), если произвести замену $c_{ij}\leftarrow c^{*}-c_{ij},\ i,j=\overline{1,n},\ c^{*}=\max\{c_{ij},\ i,j=\overline{1,n}\}$.

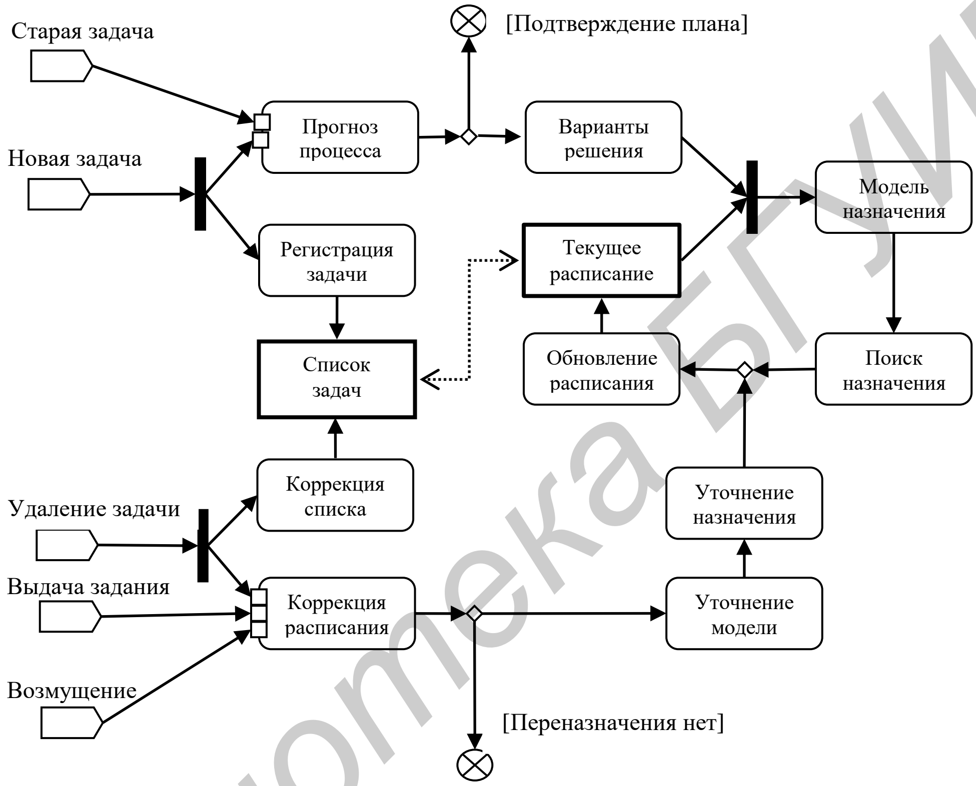

Рис. 1. Схема распределения ресурсов для обслуживания заявок на решение задач

Представляющие практический интерес задачи с прямоугольной матрицей

$$\min\left\{\sum_{i=1}^{m}\sum_{j=1}^{n}c_{ij}\cdot x_{ij}\,\middle|\,\sum_{i=1}^{m}x_{ij}=\sum_{j=1}^{n}x_{ij}=1;\ x_{ij}\ge 0;\ i=\overline{1,m},\ j=\overline{1,n}\right\} \qquad (3)$$

могут рассматриваться как задачи вида (2) после дополнения матрицы строками или столбцами с нулевыми элементами. Таким образом, ЛЗН можно рассматривать в постановке (3), когда $m\le n$. Случай $m>n$ легко сводится к решению (3) после транспонирования матрицы и соответствующей замены содержательной интерпретации индексов строк и столбцов [1,5].

Постановка задачи

Реальные процессы решения задач агентами характеризуются наличием интервала от момента появления задачи до наиболее позднего момента начала ее решения. В этом интервале можно пересматривать альтернативы решения, по крайней мере, в механотронных системах. Очевидно, что максимум эффективности управления достигается при минимуме задержек на решение ЛЗН. Предлагается учесть дискретность процесса формирования портфеля заявок, явно используя понятия наиболее раннего и позднего срока начала решения задачи для упреждающего поиска окончательного назначения. Очевидно, что если процедура назначения дополняет граф оптимального паросочетания и агентов при поступлении новых заявок, то задержка времени определяется сложностью обработки последней группы заявок.

Вычислительная сложность решения ЛЗН имеет полиномиальную зависимость от параметров матрицы [1]. Вид зависимости определяется методом решения и схемой его реализации. Практически интересные методы рассматривают ЛЗН как специальный случай задачи линейного программирования. Среди методов решения выделяют прямые методы, прямо-двойственные и комбинированные. Теоретические и экспериментальные исследования эффективности решения ЛЗН отражены во многих источниках [1]. Часто упоминаемая в публикациях оценка вычислительной сложности относится к исторически первому венгерскому методу имеет вид от $O(n^4)$ до $O(n^3)$ [1]. Процесс поиска назначения позже было предложено рассматривать в терминах поиска кратчайших путей приращений потенциалов. При этом учет возможностей реализации схемы Дейкстры для поиска кратчайших путей позволил получить асимптотики от $O(n^{3/4}m \cdot \log c^*)$ до $O(\sqrt{n}m \cdot \log(nc^*))$. Последнюю оценку имеет алгоритм метода кратчайшего пополняющего пути (Shortest Augmenting Path, SAP) [6]. Другой вариант поиска назначения реализует так называемый аукционный алгоритм, вычислительная сложность которого также $O(\sqrt{n}m \cdot \log(nc^*))$ [4].

Очевидно, что в случае решения потока ЛЗН вместо наивной схемы независимой обработки отдельных задач можно использовать пересчет задачи (2) после изменения некоторых элементов строки матрицы. Вычислительная сложность пересчета одной строки снижается в n раз, если использовать наследование результатов предшествующего решения.

Таким образом, представляет интерес построение алгоритмов получения решения новой ЛЗН на основе наследуемых результатов, что позволит повысить быстродействие системы координации. Известный прототип рассматриваемой задачи – задача о динамическом назначении [3,4,6]. В [6] рассматривается расширение статического варианта ЛЗН для случая, когда меняются элементы матрицы стоимости и предложен алгоритм решения на основе венгерского метода. Далее будет рассмотрено улучшение подобного подхода на основе метода SAP.

Алгоритм решения задачи

Решение классических открытых ЛЗН, записываемых в виде (3) обычно есть вектор назначений строк матрицы коэффициентов ее столбцам

$$R = \left\{ r_j = i \,\middle|\, x_{ij} = 1,\ i = \overline{1,m},\ j = \overline{1,n} \right\}. \qquad (4)$$

Известно, что наиболее эффективные для решения задачи (3) алгоритмы венгерского метода строятся с учетом особенностей двойственной задачи

$$\max\left\{ \sum_{i=1}^{m} u_i + \sum_{j=1}^{n} v_j \mid c_{ij} - u_i - v_j \ge 0,\ \ i = \overline{1,m},\ j = \overline{1,n} \right\}. \qquad (5)$$

Предлагается для оптимизации динамического назначения (1) модифицировать инкрементальный алгоритм венгерского метода [4,6] с целью многократного обновления с вычислительной сложностью $O(n^2)$ текущего паросочетания для новой или изменяемой строки $i,\ i \in \overline{1,m}$. Обозначим для каждой вершины x графа множество непосредственно достижимых смежных выходных вершин x', $x' = \{y \mid x \to y\}$, а входных вершин – $'x$, $'x = \{x \mid x \to y\}$, где $x \to y,\ x, y \in N$.

Предлагаемый инкрементальный алгоритм итераций метода SAP, представляемый функцией $slap(i)$, реализует обработку последствий изменения весов выходных связей любой строки (рис. 2).

```
function slap(i) {
  foreach j ∈ i' { p_j = i, d_j = c_ij − v_j; }
  R = ∅, S = ∅, T = {i'};
  while (true) {
   if (|S| = 0) {
    h = min{ d_j | j ∈ T};
    S = { j | (d_j = h) ∧ ( j ∈ T)}; T ← T \ S;
    foreach j ∈ S if (r_j = 0) go to back;
   }
   k = S_1, S ← S \ {k}, R ← R ∪ {k}, l = r_k;
   foreach j ∈ T if (h + c_lj − v_j < d_lj) {
     d_j = h + c_lj − v_j; p_j = l;
     if (d_j = h) {
      if (r_j = 0) go to back;
      S ← S ∪ { j}, T ← T \ { j};
     }
   }
  back: foreach k ∈ R v_k ← v_k + d_k − h;
  do {l = p_j; r_j = l; k = j; j = q_l; q_l = k; } while (i ≠ l);
 }
}
```

Рис. 2. Алгоритм итерации решения ЛЗН методом SAP

Такая функция обеспечивает формирование вектора назначения строк при их последовательном рассмотрении (рис. 3), а также корректное изменение вектора назначения в режиме реоптимизации [2,4]. Ее схема не меняется и для ЛЗН, формулируемой в виде графа.

```
function SLAP() {
  foreach j ∈ 1,n {r_j = 0, v_j = 0; }
  foreach i ∈ 1,m slap(i);
}
```

Рис. 3. Алгоритм решения отрытой ЛЗН методом SAP

В отличие от алгоритма итерации назначения венгерским методом, развитие дерева паросочетаний здесь проводится алгоритмом Дейкстры, где классификация состояния вершин дерева проводится экономным способом. Преимущества метода SAP особенно заметны для разреженных графов.

Структура данных представления динамической задачи

В случае задач динамического назначения параметры m и n зависят от времени, поэтому узким местом становятся операции выделения подлежащих реоптимизации строк и столбцов. Пусть в реальном времени формируется поток ЛЗН с номерами $k = 1,2,...$, а на каждом шаге изменяется $T(k)$ элементов матрицы $\left\| c_{ij}, \; i = \overline{1,m}, \; j = \overline{1,n} \right\|$. Первоначально пусть $c_{ij} = \infty, \; i = \overline{1,m}, \; j = \overline{1,n}$, тогда появление элементов $c_{ij} < \infty$ означает наличие дуги $i \to j$ графа назначения. Прямолинейный поиск изменившихся строк в матрицах ЛЗН с номерами k и $k+1$ характеризуется вычислительной сложностью $O(mn)$.

Для быстрого отображения изменений матриц предлагается использовать векторы $X^k(i) = k \cdot (c_{ij}^k \equiv c_{ij}^{k-1})$ и $Y^k(j) = k \cdot (c_{ij}^k \equiv c_{ij}^{k-1})$, $i \in \overline{1,m}, \; j \in \overline{1,n}, \; k > 0.$ Индексы измененных строк и столбцов будем сохранять в стеках индексов строк $H_x^k(t)$, $|H_x^k(t)| = h_x^k(t),$ и столбцов

$H_y^k(t)$, $|H_y^k(t)| = h_y^k(t)$, на шаге t формирования матрицы на этапе $k+1$, когда $t \in \overline{1, T(k+1)}$.
Если $X^0(i) = 0,\ i = \overline{1, m},\ Y^0(j) = 0,\ j = \overline{1, n}$, а $c_{ij}^0 = \infty,\ i = \overline{1, m},\ j = \overline{1, n}$, то операция учета фактов $c_{ij}^{k+1} \leftarrow c_{ij}^k$ на этапе $k+1$ имеет вид:

$$
\begin{cases}
h_x^{k+1}(0) = 0;\ h_y^{k+1}(0) = 0; \\
\forall\ (t \in \overline{1, T(k+1)}) \wedge (i \subset \overline{1, m}) \wedge (j \subset \overline{1, n})\ \ if\ \ (c_{ij}^{k+1}(t) \neq c_{ij}^k)\ \{ \\
\quad \text{if}\ \ (X^{k+1}(i) \neq k+1)\ \{ \\
\quad\quad h_x^{k+1}(t) \leftarrow h_x^{k+1}(t) + 1,\ H_x^{k+1}(h_x^{k+1}(t)) \leftarrow i,\ X^{k+1}(i) \leftarrow k+1; \\
\quad \} \\
\quad \text{if}\ \ (Y^{k+1}(j) \neq k+1)\ \{ \\
\quad\quad h_y^{k+1}(t) \leftarrow h_y^{k+1}(t) + 1,\ H_y^{k+1}(h_y^{k+1}(t)) \leftarrow j,\ Y^{k+1}(j) \leftarrow k+1; \\
\quad \} \\
\quad c_{ij}^{k+1} \leftarrow c_{ij}^{k+1}(t); \\
\}
\end{cases}
\tag{6}
$$

Сложность операции изменения элемента очередной матрицы $- O(1)$. Очевидно, что истинность выражения $h_x^k(k+1) > h_y^k(k+1)$ означает необходимость решения ЛЗН с заменой строк столбцами.
Алгоритм функции $slap(i)$ исключает необходимость дополнения матрицы C строками или столбцами нулевыми элементами, но требует выполнения условия $m \leq n$. Формальное инвертирование матриц выполняется одношаговой заменой адресной функции доступа к элементам матрицы C.

Фильтрация несущественных событий

Появление новых элементов множеств вершин не обязательно приводит к изменению существующего оптимального паросочетания. Контроль устойчивости назначения, обеспечивающий грубость системы к незначительным возмущениям элементов или размерности матрицы, предлагается реализовать на основе результатов решения задача оценки интервалов устойчивости решений ЛЗН:
для каждого элемента c_{ij} в (2) необходимо найти интервал $I(i, j) = (a_{ij}, b_{ij})$, в котором значения таких элементов могут быть изменены без нарушения структуры оптимального назначения (3).
Пусть граф оптимального паросочетания (текущего решения ЛЗН) представлен элементами $E_m = \{(r_j, j) \big| (r_j \leq m),\ j = \overline{1, n}\}$, а оставшиеся элементы матрицы образуют
$E_u = \{(i, j),\ i = \overline{1, m},\ j = \overline{1, n}\} \setminus E_m$. Очевидно, что $|E_m| = m$, а $|E_u| = m \cdot (n-1)$, что приходится учитывать при оценке вычислительной сложности алгоритма.
Обозначим $\delta_m(x, y)$ и $\delta_u(x, y)$ – допустимое изменение веса любого ребра $x \to y$, где $(x, y) \in E_m \cup E_u$. Интервалы значений веса ребра, в которых назначение ребра остается неизменным, для задачи минимизации (1) определяются из элементарных рассуждений:
$I_{\min}^m(x, y) = (-\infty, c_{xy} + \delta_m(x, y)],\ \ (x, y) \in E_m;$
$I_{\min}^u(x, y) = [c_{xy} - \delta_u(x, y), +\infty),\ \ (x, y) \in E_u.$
Будем считать, что ребро графа оптимального паросочетания скрыто, если его вес увеличен так, что ребро больше не является частью существующего решения. Скрытие реализуется назначением веса из интервала $(c_{xy} + \delta_m(x, y), +\infty)$.

Известно, что для элементов оптимального паросочетания справедливо $c_{xy} = u_x + v_y$, $(x, y) \in E_m$. Пусть оценка оптимального решения ЛЗН есть

$$Z^0 = \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij} \cdot x_{ij}^0 = \sum_{i=1}^{m} u_i^0 + \sum_{j=1}^{n} v_j^0 . \qquad (7)$$

Нетрудно показать, что если скрыто ребро $x \to y$, $(x, y) \in E_m$, то процесс реоптимизации, начинающийся в вершине x, завершится в вершине y, потенциал которой не изменится. Меняется только потенциал строки u_x, а в соответствии с выражением (7): $u_x^1 - u_x^0 = Z^1 - Z^0$, $x \in \overline{1,m}$. Здесь нулевой верхний индекс использован для пометки исходного, а единичный – нового решения.

Отсюда следует, что $\delta_m(x, y) = Z^1 - Z^0$. Интервал безопасного изменения значений веса скрываемого ребра определяется выражением

$$I_{\min}^m(x, y) = (-\infty, c_{xy} + Z^1 - Z^0] = (-\infty, c_{xy} + u_x^1 - u_x^0], \ (x, y) \in E_m . \qquad (8)$$

Использование разности потенциалов исключает необходимость прямолинейного вычисления оценок решений задачи (1), требующего $m + n$ шагов.

Подобным образом рассматриваются ребра, не принадлежащие оптимальному решению, когда $c_{xy} \ge u_x + v_y$, $(x, y) \in E_u$. Если вес таких ребер менять в интервале $(u_x + v_y, +\infty)$, то структура решения (2) остается неизменной.

Пусть $\varepsilon_u(x, y) = c_{xy} - \delta_u(x, y)$. Очевидно, что ребро графа паросочетания открыто, если его вес уменьшен так, что это ребро становится частью нового оптимального решения. Открытие ребра наступает в случае, когда $c_{xy} \leftarrow \varepsilon_u(x, y)$. Для поиска $\varepsilon_u(x, y)$ предлагается воспользоваться выражением (8), инвертируя направление шагов процесса построения интервала. Конечная граница интервала $I_{\min}^m(x, y)$ станет начальной границей интервала $I_{\min}^u(x, y)$. Нулевой шаг в (8) становится решением ЛЗН для гарантированно приводящего к открытию ребра значения $c_{xy} = -\infty$, а единичный шаг соответствует решению ЛЗН с исходной матрицей. В результате получаем

$$I_{\min}^u(x, y) = (-\infty + Z^0 - Z^1, +\infty] = (-\infty + u_x^0 - u_x^1, +\infty], \ (x, y) \in E_u . \qquad (9)$$

Оценка выражения (9) требует лишь одношаговой реоптимизации исходной задачи (1).

Вычислительная сложность реализации выражений (8) и (9) – $O(n^4)$ (при использовании венгерского метода). Однако следует учитывать, что:

реальный интерес представляет оптимальное паросочетание, а так как $|E_m| = m,$ то вычислительная сложность реализации (8) – $O(mn^2)$;

вычислительная схема реализации выражений (8) и (9) не исключает возможности использования метода SAP для снижения сложности реализации (8) до $O(\sqrt{n} \cdot \log(nc^*))$;

вычисление выражений (8) и (9) может быть прервано, проводится с приоритетом оценки для вновь измененных строк и выполняться в фоновом режиме.

Таким образом, определение интервалов устойчивости естественным образом согласуется с последовательной схемой выражения (1).

Заключение

Представленные результаты позволяют обеспечить мониторинг потока задач в системах взаимодействующих агентов, обеспечивая минимальные задержки на обработку особых событий и откладывание принятие решений на максимально возможный срок. Сокращение времени получения оптимального решения ЛЗН в первом приближении пропорционально объему локальных изменений данных задачи. Дополнительная память для хранения наследуемых значений потенциалов строк и столбцов не превышает объема $O(m+n)$. Структура оптимального паросочетания при решении задач динамического назначения может оставаться без изменения при незначительном изменении значений весовых коэффициентов. Выявление областей допустимого изменения таких коэффициентов без нарушения структуры решения позволяет обеспечить отказ от итерации реоптимизации. При этом сокращается потребность в информационном взаимодействии между агентами.

Список литературы / References

Spivey M.Z., Powel W.B. The Dynamic Assignment Problem// Transportation science, vol. 38(4), 2004. P. 399-419.

Pentico D.W. Assignment problems: A golden anniversary survey// European Journal of Operational Research, vol. 176(2), 2007. P. 774-793.

Powell W.B., Snow W. Adaptive Labeling Algorithms for the Dynamic Assignment Problem//Transportation Science, Vol. 34, No. 1. – February 2000. P. 50-66

Toroslu I.H., Üçoluk G. Incremental assignment problem //Information Sci., vol. 177, 2007. P. 1523-1529.

Bijsterbosch J., Volgenant A. Solving the Rectangular assignment problem and applications//Annals of Operations Research, vol. 181(1), 2010. P. 443-462.Jonker R., Volgenant A. A shortest path algorithm for dense and sparse linear assignment problem// Computing, vol. 38, 1987. P. 325-340.

The Dynamic Hungarian Algorithm for the Assignment Problem with Changing Costs/G.A. Korsah [et al.] //Tech. report CMU-RI-TR-07-27[Electronic resource]: Robotics Institute, Carnegie Mellon University, July, 2007. – Mode of access: https://www.ri.cmu.edu/pub_files/pub4/mills_tettey_g_ ayorkor_2007_3/ mills_tettey_g_ayorkor_2007_3.pdf. – Date of access: 24.08.2017.

Сведения об авторах
Ревотюк М.П., к.т.н., доцент,
доцент кафедры информационных технологий автоматизированных систем Белорусского государственного университета информатики и радиоэлектроники.

Кот О.В., м.т.н., аспирант кафедры информационных технологий автоматизированных систем Белорусского государственного университета информатики и радиоэлектроники

Адрес для корреспонденции
220013, Республика Беларусь,
г. Минск, ул. П. Бровки, д. 6,
Белорусский государственный университет информатики и радиоэлектроники
тел. +375-17-293-86-58;
e-mail: rmp@bsuir.by
Ревотюк Михаил Павлович

Information about the authors
Revotjuk M.P., PhD, associate professor,
associate professor of the information technologies in automated systems department of Belarusian state university of informatics and radioelectronics.

Kot O.V., M. Sci., PG student of the information technologies in automated systems department of Belarusian state university of informatics and radioelectronics.

Address for correspondence
220013, Republic of Belarus,
Minsk, P. Brovka st., 6,
Belarusian state university of informatics and radioelectronics
tel. +375-17-293-86-58;
e-mail: rmp@bsuir.by
Revotjuk Mikhail Pavlovich