УДК 51-73;64-83;64.069.8

ВЕКТОРНАЯ ОПТИМИЗАЦИЯ С ПРОГРАММИРОВАНИЕМ В СРЕДЕ *MATHCAD* ПРИ КОМПЛЕКСИРОВАНИИ МАШИН И АГРЕГАТОВ

В.А. Богатырев[1], К.А. Булыгин[2], В.Ю. Пинкевич[3]

[1]*Санкт-Петербургский государственный университет сервиса и экономики (СПбГУСЭ),*
191015, *Санкт-Петербург, ул. Кавалергардская,* 7
[2,3]*Национальный исследовательский университет информационных технологий, механики и оптики.* 197101*, Санкт-Петербург, пр. Кронверкский,* 49

Аннотация – Рассматриваются возможности использования средств программирования в системе математических расчетов *Mathcad* для создания набора функций, направленного на поддержку решения задач целочисленной векторной оптимизации. В качестве объекта оптимизации рассматривается комплекс машин и агрегатов, предусматривающих многоэтапный процесс выполнения работ.

Ключевые слова: целочисленной векторной оптимизации, Mathcad, программирование, аддитивный критерий, надежность, среднее время пребывания.

VECTOR OPTIMISATION WITH PROGRAMMING IN THE ENVIRONMENT OF *MATHCAD* AT КОМПЛЕКСИРОВАНИИ CARS AND UNITS

V.A.Bogatyrev, Bulygin K.A, Pinkevich V. J
St.-Petersburg state university of service and economy (SPbSUSE),
191015, *St.-Petersburg, street Kavalergardsky,*7, *lit. A*
National Research University of Information Technologies, Mechanics and Optics
197101, *St.-Petersburg, avenue Kronverksky* 49

Summary – Possibilitiesof use of means of programming in system of mathematical calculations Mathcad for the creation of a set of functions directed on support of the decision of problems of integervector optimisation. As object of optimisation the complex of cars and the units providing многоэтапный process of performance of works**.**

Keywords: integer vector optimisation, Mathcad, programming, additive criterion, reliability, average time of stay.

Введение

Проектирование машин, агрегатов и их комплексирование в системы требует решения задачи оптимизации структуры с целью максимизации надежности и производительности системы при минимизации затрат на ее построение и эксплуатацию.

Задача оптимизации является векторной, её сложность обусловлена тем, что одно решение (альтернатива) может превосходить другую по одним показателям и уступать по другим [1].

В статье рассматривается задача векторной оптимизации при построении резервированных комплексов машин и агрегатов, реализующих многоэтапный процесс выполнения работ [2,3].

Эффективность поиска наилучших решений по реализации структуры исследуемых многоуровневых систем во многом зависит от выбора инструментальных средств оптимизации и эффективности их использования.

Решение оптимизационной задачи с использованием встроенных функций *Minimize*(*f*, *x*1, *x*2,…)/*Maximize*(*f*, *x*1, *x*2,…) системы *Mathcad* сопряжено с трудностями получения целочисленного решения и определения параметров оптимизации над знаком суммы целевой функции-[2-11].

В предлагаемой статье рассматриваются возможности использования средств программирования в системе математических расчетов *Mathcad* для со-

здания набора функций, направленного на поддержку решения задач целочисленной векторной оптимизации.

Постановка задачи оптимизации

В качестве объекта оптимизации рассмотрим комплекс машин и агрегатов, предусматривающих многоэтапный процесс выполнения работ. Для повышения надежности и производительности системы предусматривается резервирование машин, реализующих различные этапы процесса обработки.

Требуется найти число (кратность резервирования) машин, реализующих каждый этап обработки $(m_1, m_2, ..., m_M)$, при котором обеспечивается максимум надежности системы, минимум среднего времени пребывания в ней запросов, а стоимость реализации системы не должна превышать выделенных на это средств:

$$P(m_1, m_2, ..., m_M) \to \max$$

$$T(m_1, m_2, ..., m_M) \to \min, \text{ при}$$

$$C(m_1, m_2, ..., m_M) = \sum_{i=1}^{M} m_i c_i \le C_0 .$$

Возможна также другая постановка задачи. Требуется найти число машин, реализующих каждый этап обработки, при котором стоимость реализации системы минимальна и обеспечиваются требуемые технические показатели системы:

$$C(m_1, m_2, ..., m_M) \to \min, \text{ при}$$

$$P(m_1, m_2, ..., m_M) \ge P_0 \text{ и}$$

$$T(m_1, m_2, ..., m_M) \le T_0 .$$

При этом T_0, P_0, C_0 – предельно допустимые значения среднего времени реализации процесса обработка, надежности и стоимости системы.

При оптимизации будем считать заданными:

- интенсивность входного потока запросов λ;
- показатели надежности машин (вероятности безотказной работы) $p_1, p_2, p_3, ..., p_M$;
- средние времена выполнения запросов машинами на этапах обработки $v_1, v_2, v_3, ..., v_M$;
- стоимости машин, выполняющих соответствующие этапы процесса обработки $c_1, c_2, c_3, ..., c_M$.

Решение рассматриваемых задач векторной оптимизации, как правило, включает поиск области Парето и выработку решающего правила, основанного на компромиссе между значениями компонент векторного показателя.

Оценка частных показателей качества

Возможности системы по обработке запросов определим по среднему времени пребывания запросов в системе и по ее надежности [4, 10]. В простейшем случае каждый узел представим системой массового обслуживания типа *M/M/1*. Считая, что каждый запрос последовательно обслуживается одним из исправных узлов на каждом уровне системы, среднее время пребывания запросов в системе оценим как:

$$T(m_1, m_2, m_3, ...m_M) = \sum_{i=1}^{M} \frac{v_i}{1 - v_i \lambda / m_i} ,$$

где $(m_1, m_2, m_3, ..., m_M)$ – число узлов на каждом уровне.

Надежность системы оценим по вероятности сохранения работоспособности, которая зависит от формулировки условия отказа (сохранения функционирования *Y*).

Если система работоспособна, когда исправно не менее чем d_i узлов на *i*-м уровне системы, то надежность системы оценим как [5, 6]:

$$P(m_1, m_2, m_3, ..., m_M) = \sum_{k_1=d_1}^{m_1} \sum_{k_2=d_2}^{m_2} \sum_{k_3=d_3}^{m_3} ... \sum_{k_M=d_M}^{m_M} C_{m_1}^{k_1} C_{m_2}^{k_2} C_{m_3}^{k_3} ... C_{m_M}^{k_M} \times$$

$$\times p_1^{k_1} p_2^{k_2} p_3^{k_3} ... p_M^{k_M} (1-p_1)^{m_1-k_1} (1-p_2)^{m_2-k_2} (1-p_3)^{m_3-k_3} ... (1-p_M)^{m_M-k_M} ,$$

где $C_{m_i}^{k_i} = m_i!/k_i!(m_i - k_i)!$

Особенность представленной функции заключается в том, что искомые параметры оптимизации находятся над знаком суммы, что затрудняет использование встроенных функций оптимизации *Minimize*/*Maximize* системы *Mathcad* [5,11]. Программирования функций поиска целочисленного оптимума отдельных критериев ранее рассматривалось в работах [6, 11], в представленной же статье предлагается взаимосвязанный набор функций поддержки решения задач целочисленной векторной оптимизации .

Функции оценки частных показателей качества в системе *Mathcad*

Определим функцию для расчёта времени пребывания заявки в системе [12], интерпретируя каждый узел исследуемой структуры системой массового обслуживания типа *M/M/1*, как:

$$T(\lambda,k) := \left| \begin{array}{l} r \leftarrow 0 \\ for\ i \in 0..last(k) \\ \quad \left| \begin{array}{l} \rho \leftarrow \dfrac{\lambda}{k_i} \cdot b_i \\ return\ \infty\ if\ \rho \geq 1 \\ r \leftarrow r + \alpha_i \cdot \dfrac{b_i}{1-\rho} \end{array} \right. \\ r \end{array} \right. .$$

Эта функция принимает два аргумента: интенсивность входного потока заявок и вектор числа элементов на каждом уровне. Кроме этого, она учитывает значения вектора α, определяющего коэффициенты передач (сколько раз заявка попадает в узел каждого типа). Функция для расчёта вероятности работоспособного состояния системы включает вычисление числа сочетаний из *n* по *k* (для этого используется треугольник Паскаля):

$$Combins := pascal\_triangle(\max(m))$$
$$(Combins_{k,n} = C_n^k)$$

$$P(N) := \left| \begin{array}{l} r \leftarrow 1 \\ for\ j \in 0..last(p) \\ \quad \left| \begin{array}{l} C \leftarrow Combins^{\langle N_j \rangle} \\ r \leftarrow r \cdot \left[\sum_{i=s_j}^{N_j} \left[C_i \cdot (p_j)^i \cdot (1-p_j)^{N_j - i} \right] \right] \end{array} \right. \\ r \end{array} \right. .$$

Функции для нахождения интенсивности максимального входного потока, выдерживаемого системой, и её стоимости имеют вид:

$$\Lambda(N) := \min\left(\frac{N}{b}\right),\ C(N) := N \cdot c .$$

Функция, формирующая множество допустимых систем, из которых потом выбирается наилучшая со вспомогательными функциями:

$$Crits(N) := stack(P(N),\ T(\lambda, N),\ \Lambda(N),\ C(N),\ format("\{0\},\{1\},\{2\}",\ N_0, N_1, N_2))$$

$$is\_valid(Crits) := (Crits_0 \geq P0) \wedge (Crits_1 \leq T0) \wedge (Crits_3 \leq C0)$$

$$RN := \left| \begin{array}{l} i \leftarrow 0 \\ for\ n_0 \in s_0..m_0 \\ \quad for\ n_1 \in s_1..m_1 \\ \quad\quad for\ n_2 \in s_2..m_2 \\ \quad\quad\quad \left| \begin{array}{l} N \leftarrow stack(n_0, n_1, n_2) \\ Crits\_N \leftarrow Crits(N) \\ if\ is\_valid(Crits(n) \\ \quad \left| \begin{array}{l} RN^{\langle i \rangle} \leftarrow Crits\_N \\ i \leftarrow i + 1 \end{array} \right. \end{array} \right. \\ RN^T \end{array} \right.$$

Здесь *Crits*(*N*) и *is_valid*(*Crit*) – вспомогательные функции, которые используются при вычислении значения матрицы *RN*.

Crits(*k*) вычисляет значения всех критериев данной системы (которая задаётся вектором *k* – число элементов на каждом уровне), в том числе имя системы (которое будет критерием для лица, принимающего решение) в виде строки.

is_valid(*Crit*) принимает вектор критериев и проверяет, удовлетворяют ли критерии заданным условиям (возвращает 1 или 0 – «да» или «нет»).

Матрица *RN* создаётся при помощи вложенных циклов *for* (число циклов соответствуе числу уровней системы) и состоит из векторов *k* (то есть системы) с числом устройств на каждом уровне от необходимого (здесь «1») до максималь-

ного, которое можно использовать, не превысив бюджет (ограничение *C0*). Далее при помощи вспомогательных функций для каждой системы вычисляются значения критериев и проверяется, удовлетворяют ли эти значения ограничениям. При удовлетворении весь вектор критериев помещается в матрицу *RN* – матрицу допустимых решений. В итоге формируется матрица, строками которой являются вектора критериев (*Results*) с именами решений (*Names*).

Дальше определяются вспомогательные функции для отделения от этой матрицы столбца (вектора) имён и собственно матрицы критериев, которыми оперируют функции методов многокритериальной оптимизации.

$$rn2n(RN) := RN^{\langle 4 \rangle};$$

$$rn2r(RN) := submatrix(RN, 0, rows(RN) - ... \\ ... - 1, 0, cols(RN) - 2).$$

Функции нахождения области Парето

Для нахождения области Парето определим следующие вспомогательные функции:

$$is\_member(z, V) := \left| \begin{array}{l} for\ i \in 0..last(v) \\ \quad return\ 1\ if\ V_i = z \\ 0 \end{array} \right.$$

$$is\_less\_better(i\_crit) := (i\_crit = 1) \vee (i\_crit = 3)$$

$$fill\_vector(value, length) := \left| \begin{array}{l} for\ i \in 0..length - 1 \\ \quad V_i \leftarrow value \\ V \end{array} \right.$$

Функция *is_member*(*z*, *V*) служит для проверки, является ли значение *z* элементом множества *V* (заданного в виде вектора). *is_less_better* возвращает 1 («истина»), если критерий, индекс которого передается как параметр, имеет смысл «чем меньше, тем лучше», иначе возвращается 0 («ложь»): *P* и Λ мы максимизируем, а *T* и *C* – минимизируем. *fill_vector* возвращает вектор указанной длины, заполненный нужным значением.

Функция *normalize* нормализует значения по столбцам, поскольку в них находятся значения одного и того же критерия для разных систем.

$$normalize(R) := for\ i \in 0..cols(R) - \left| \begin{array}{l} C \leftarrow R^{\langle i \rangle} \\ Cmin \leftarrow min(C; \\ Cmax \leftarrow max(C; \\ C \leftarrow \left| \begin{array}{l} fill\_vector(0.5, length(C)) ... \\ ...if\ Cmin = Cmax \\ \dfrac{C - Cmin}{Cmax - Cmin}\ otherwise \end{array} \right. \\ C \leftarrow 1 - C\ if\ is\_less\_better(i) \\ R^{\langle i \rangle} \leftarrow C. \end{array} \right.$$

Далее определены функции, упрощающие нахождение области Парето:

Функция *is_pareto_worse*(*V*, *W*, *nV*, *nW*) определяет, доминирует ли по Парето вектор *W* над вектором *V*, то есть, является ли вектор *V* однозначно худшим, чем *W*.

Функция *is_pareto_bad*(*V*, *Cols*, *nV*, *N*) находит, доминирует ли по Парето над вектором *V* хоть один другой вектор, то есть лишний ли вектор *V* в матрице *Cols*.

Функции для нахождения области Парето, принимающие нормализованные значения (все критерии имеют вид «чем больше, тем лучше»):

$$is\_pareto\_worse(V,W,nV,nW) := \left| \begin{array}{l} return\ 0\ if\ \ nW = "bad" \\ is\_worse \leftarrow 0 \\ for\ \ i \in 0..last(V) \\ \quad \left| \begin{array}{l} return\ 0\ if\ \ V_i > W_i \\ is\_worse \leftarrow 1\ if\ \ V_i < W_i \end{array} \right. \\ trace("is\_pareto\_worse: \\ \quad \{0\}\ \ is\ worse\ than\ \ \{1\}", nV, nW)\ if\ \ is\_worse \\ is\_worse. \end{array} \right.$$

$$is\_pareto\_bad(V,Cols,nV,N) := \left| \begin{array}{l} for\ \ i \in 0..cols(Cols)-1 \\ \quad return\ 1\ if\ \ is\_pareto\_worse(V, Cols^{\langle i \rangle}, ... \\ ...nV, N_i) \\ 0. \end{array} \right.$$

$$pareto\_names(R,N) := \left| \begin{array}{l} Cols \leftarrow R^T \\ i\_good \leftarrow 0 \\ \\ for\ i \in 0..cols(Cols)-1 \\ \quad \left| \begin{array}{l} V \leftarrow Cols^{\langle i \rangle} \\ N_i \leftarrow "bad"\ \ if\ \ is\_pareto\_bad(V,Cols,N_i,N) \\ otherwise \\ \quad \left| \begin{array}{l} N\_good_{i\_good} \leftarrow N_i \\ i\_good \leftarrow i\_good+1 \end{array} \right. \end{array} \right. \\ N\_good. \end{array} \right.$$

$$filter\_names(R,N,N\_new) := ...$$

$$... = \left| \begin{array}{l} Cols \leftarrow R^T \\ i\_new \leftarrow 0 \\ for\ i \in 0..last(N) \\ if\ is\_member(N_i, N\_new) \\ \quad \left| \begin{array}{l} Cols\_new^{\langle i\_new \rangle} \leftarrow Cols^{\langle i \rangle} \\ i\_new \leftarrow i\_new+1 \end{array} \right. \\ Cols\_new^T \end{array} \right. .$$

Функция *pareto_names*(*R*, *N*) по заданной матрице критериев (*R*) и вектору имен систем (*N*) отбирает в новый вектор имена систем, входящих в область Парето. Функция *filter_names*(*R*, *N*, *N_new*) по этому вектору составляет матрицу критериев ("*R_new*") систем, входящих в область Парето.

Функции для решения задач векторной оптимизации

Для нахождения оптимального решения при векторной оптимизации могут использоваться: минимаксный критерий; аддитивный критерий; мультипликативный критерий; метод отклонения от идеала; метод главного критерия; метод последовательных уступок и др.

Приведём определение функции для нахождения оптимального решения по аддитивному критерию.

$$additive(R, N, W) := \left| \begin{array}{l} Cols \leftarrow R^T \\ max\_value \leftarrow -\infty \\ name \leftarrow "" \\ for\ i \in 0..cols(Cols) - 1 \\ \quad \left| \begin{array}{l} value \leftarrow W \cdot Cols^{\langle i \rangle} \\ \\ if\ value > max\_value \\ \quad \left| \begin{array}{l} max\_value \leftarrow value \\ name \leftarrow N_i \end{array} \right. \end{array} \right. \\ name \end{array} \right.$$

Функция требует нормализованных значений, то есть в качестве аргумента *R* ей должна передаваться матрица критериев в области Парето, обработанная функцией *normalize*. Кроме того, в функцию передаётся вектор имён систем, которым принадлежат эти критерии, и вектор весов критериев *W*. Для удобства матрица критериев *R* представляется так, чтобы каждый столбец содержал характеристики одной системы. Функция скалярно умножает каждый столбец (вектор критериев системы) на вектор весов, получая значение аддитивного критерия для данной системы. После этого полученное значение аддитивного критерия сравнивается с максимальным на данный момент значением и, если новое значение больше максимального, то оно само становится максимальным, а его имя запоминается как возможный ответ. После перебора всех систем в переменной *name* оказывается имя системы с максимальным значением аддитивного критерия.

Здесь *trace* – встроенная функция *Mathcad*, которая выводит в окно отладки строку с нужными переменными при включенной опции *Tools* > *Debug* > *Toggle debugging*.

Пример оптимизации

Начальные условия задачи можно задать в таком виде:

b := stack(1, 4, 2) – время обслуживания;

p := stack(0.99, 0.9, 0.92) – надёжность;

c := stack(1, 5, 6) – стоимость.

После генерации матрицы с характеристиками решений и применения к ней функций нормализации и области Парето, можно найти параметры оптимальных систем (число машин на каждом уровне системы) при использовании разных методов:

$$RateNames := \begin{pmatrix} minimax(R\_pn, N\_pn) \\ additive(R\_pn, N\_pn, Weights) \\ multiplicative(R\_pn, N\_pn, Weights) \\ divergent(R, N, Weights) \\ main\_crit(R, N) \\ seq\_concession(R\_pn, N\_pn) \\ stem(R\_pn, N\_pn, stem\_mins) \end{pmatrix} = \begin{pmatrix} "2,8,4" \\ "4,6,3" \\ "4,6,3" \\ "1,6,5" \\ "1,4,2" \\ "3,3,3" \\ "3,4,2" \end{pmatrix}.$$

Определим значения критериев у выбранных систем, чтобы можно было сравнить их и принять окончательное решение. Для этого определим функцию, возвращающую матрицу критериев систем по вектору имён этих систем.

$$names2crits(Names, R, N) := \ldots$$

$$\ldots = \left| \begin{array}{l} Cols \leftarrow R^T \\ for\ i \in 0..cols(Cols) - 1 \\ \quad \left| \begin{array}{l} j \leftarrow match(Names_i, N)_0 \\ Crits^{\langle i \rangle} \leftarrow Cols^{\langle j \rangle} \end{array} \right. \\ Crits^T. \end{array} \right.$$

Получим характеристики выбранных систем:

$$names2crits(RateNames,\ R,\ N) := \begin{pmatrix} P & T & \Lambda & C & \\ 1 & 8.889 & 2 & 66 & \text{минимаксный} \\ 0.999 & 9.028 & 1.5 & 52 & \text{аддитивный} \\ 0.999 & 9.028 & 1.5 & 52 & \text{мультиплик ативный} \\ 0.99 & 9.289 & 1 & 61 & \text{отклонения от идеала} \\ 0.984 & 10 & 1 & 33 & \text{главного критерия} \\ 0.998 & 9.905 & 0.75 & 36 & \text{последовательной уступки} \\ 0.993 & 9.643 & 1 & 35 & STEM \end{pmatrix}$$

Заключение

Таким образом, предложен набор функций для решения оптимизационной целочисленной задачи проектирования систем резервированных машин и агрегатов в среде *Mathcad*. В отличие от использования для оптимизации встроенных в *Mathcad* функций *Minimize*/*Maximize*, предлагаемый подход позволяет найти целочисленное решение, в том числе, если искомые переменные (параметры оптимизации) находятся над знаком суммы.

Литература

1. Ларичев О.И. Теория и методы принятия решений, а также Хроника событий в Волшебных странах: Учебник. – М.: Логос, 2002. – 392 с.: ил.
2. Богатырев В.А., Богатырев С.В. Критерии оптимальности многоустойчивых отказоустойчивых компьютерных систем//научно-технический вестник СПбГУ ИТМО, 2009 №5. (Выпуск 63) с .92-98.
3. Богатырев В.А., Богатырев С.В, Лепеш Г.В. Критерии оптимальности объединения машин и агрегатов в системы //Технико-технологические проблемы сервиса № 2 .2009 с 30-35
4. Богатырев В.А., Богатырев С.В Надежность системы управления агрегатами и машинами коммунального хозяйства //Технико-технологические проблемы сервиса №4(6) 2008 с 23-27.
5. Богатырев В.А. Оценка надежности и оптимальное резервирование кластерных компьютерных систем "Приборы и системы. Управление, контроль, диагностика" №10 2006. С 18-21
6. Богатырев В.А. Оценка надежности и оптимальное резервирование кластерных компьютерных систем "Приборы и системы. Управление, контроль, диагностика" №10 2006. С 18-21
7. Богатырев В.А. Надежность функционально-распределенных резервированных структур с иерархической конфигурацией узлов // Изв. Вузов. Приборостроение. 2000. № 4. С. 67-70.
8. Богатырев В.А. Оценка вероятности безотказной работы функционально-распределенных вычислительных систем при иерархической структуре узлов // Изв. Вузов. Приборостроение. 2000. № 3.- С. 67-70.
9. Богатырев В.А. Оценка надежности функционально избыточных многомашинных вычислительных систем с реконфигурацией на основе перераспределения функций //Электронное моделирование. 1994. № 2. С. 88-90.
10. Богатырев В.А. Оценка коэффициента сохранения эффективности отказоустойчивых систем из многофункциональных модулей // Методы менеджмента качества. 2001. № 9. С. 29-33.
11. Богатырев В. А., Осипов А. В. Целочисленная оптимизация многоуровневых компьютерных систем в среде *Mathcad* //Информационные системы и технологии теория и практика вып №2 СПб. ЛТА с24-30
12. Воскобойников Ю.Е., Воскобойникова Т.Е. Программирование в математическом пакете *Mathcad* (методические указания). – Новосибирск: НГАСУ, 1999. – 32 с.

[1]*Богатырев Владимир Анатольевич, доктор технических наук, профессор кафедры Прикладных информационных технологий СПбГУСЭ, тел.*:+7 911 7260226, *e-mail :Vladimir .bogatyrev@ gmail.com;*

[2]*Булыгин Кирилл Александрович, студент кафедры Вычислительной техники СПбНИУ ИТМО, тел.* +7 921 891 69 30, *e-mail: kirill.bulygin@gmail.com;*

[3]*Пинкевич Василий Юрьевич, студент кафедры Вычислительной техники СПбНИУ ИТМО, e-mail: vass*258*@yandex.ru.*