УДК 681.51:629.78

Тестирование программных модулей расчета и контроля полетного задания на основе построения приоритетов исходных данных

Ляпин А.А.

Государственный ракетный центр имени академика В.П. Макеева,

Тургоякское шоссе,1, Миасс, Челябинская область, 456300, Россия

e-mail: lyapin-sasha@mail.ru

Аннотация

В статье предложена схема проведения автоматизированного исследовательского тестирования программного обеспечения расчета и контроля полетного задания (ПО РКПЗ). Схема тестирования реализована с учетом возможных сбоев в работе тестируемых программных модулей и основана на определении приоритетов исходных данных. Предложен способ определения приоритетов на основе показателя вычислительной устойчивости исследуемых задач. Разработанная схема позволяет уточнять область варьирования исходных данных и тем самым эффективнее использовать отведенное на этап тестирования время и получать качественную оценку работоспособности программного обеспечения расчета и контроля полетного задания.

Ключевые слова: расчет и контроль полетного задания, исследовательское тестирование, приоритеты данных, вычислительная устойчивость.

Постановка задачи

Программное обеспечение расчета и контроля полетного задания (ПО РКПЗ) является одной из ключевых систем ракетного комплекса и предназначено для подготовки данных для системы управления полетом межконтинентальной баллистической ракеты, спецавтоматики и систем управления отделяемых элементов [1,2].

ПО РКПЗ состоит из множества взаимосвязанных элементов (программных модулей) [2]. Программный модуль (далее ПМ) – функционально завершённая программная реализация частной задачи (алгоритма) РКПЗ. Под исходными данными понимается набор входных параметров программной реализации алгоритма.

С целью обеспечения качества и надежности ПО РКПЗ каждый программный модуль должен быть отлажен, отработан и протестирован в автономном режиме [1,3].

В текущей работе для отработки и тестирования ПМ из состава ПО РКПЗ предлагается использовать метод автоматизированного исследовательского тестирования [4-10], учитывающий приоритеты исходных данных, построенные на основе показателя вычислительной устойчивости исследуемой задачи.

Алгоритм исследовательского тестирования представлен на рис. 1.

Как показано на схеме, ПМ запускается в дочернем процессе, что позволяет контролировать и прерывать его работу в случае отказа и, таким образом, продолжать тестирование. Отказ ПМ – нарушение работоспособности, при котором ПМ перестает выполнять свои функции (например, зависание ПМ, зацикливание).

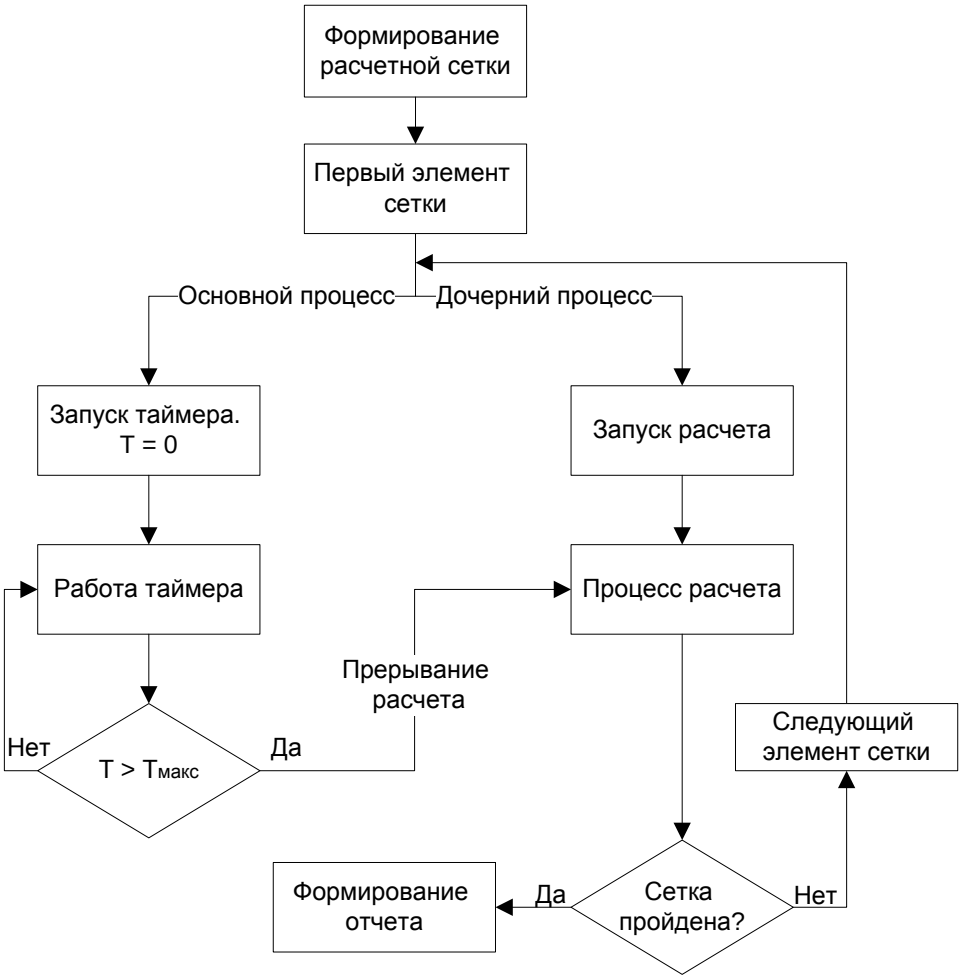

Рис. 1. Схема тестирования

Суть проведения тестирования ПМ РКПЗ заключается в построении n-мерного куба (сетки) с различными вариациями входных параметров (n равно числу контролируемых параметров) и отображение пользователю одного из срезов этого куба в виде таблицы [11].

Актуальным становится вопрос о выборе варьируемых данных и о количестве таких вариаций. С ростом сложности алгоритма и количества вариаций растут временные затраты на тестирование ПМ. Таким образом, большое значение при тестировании имеет уточнение областей варьирования исходных данных и выделение областей их изменения, наиболее важных для последующего

использования в ПМ. В данной работе такое варьирование определяется посредством построения приоритетов исходных данных тестируемого ПМ на основе показателей его вычислительной устойчивости. Приоритет – вес параметра. Задачи называются вычислительно неустойчивыми, если малые изменения исходных данных приводят к большим изменениям в решении.

Построение расчетной сетки

Пусть ПМ имеет n входных параметров $P\{p_1, p_2 .. p_n\}$, и для каждого из параметров определен приоритет $C_i = const$. Затраты на тестирование определим максимально возможным количеством тестов K_{max}.

Необходимо построить n-мерную сетку $S\{p_i^{min} \le p_i \le p_i^{max}\}$ с различным количеством вариаций параметров (в зависимости от приоритетов C_i), с учетом максимально возможного количества тестов K_{max}.

Определим количество вариаций каждого из параметров (размерность сетки) K_i, i=1..n. Стоит отметить, что в общем случае число возможных тестов $K_{cur} \neq K_{max}$. Воспользуемся алгоритмом, представленном на рис. 2.

Реализованное количество тестов:

$$K_{cur} = \prod_{i=1}^{n} K_i \, .$$

Выходные параметры алгоритма: $K_i[1..n]$ – количество узлов в n-мерной сетке для каждого параметра, K_{cur} – построенное количество тестов.

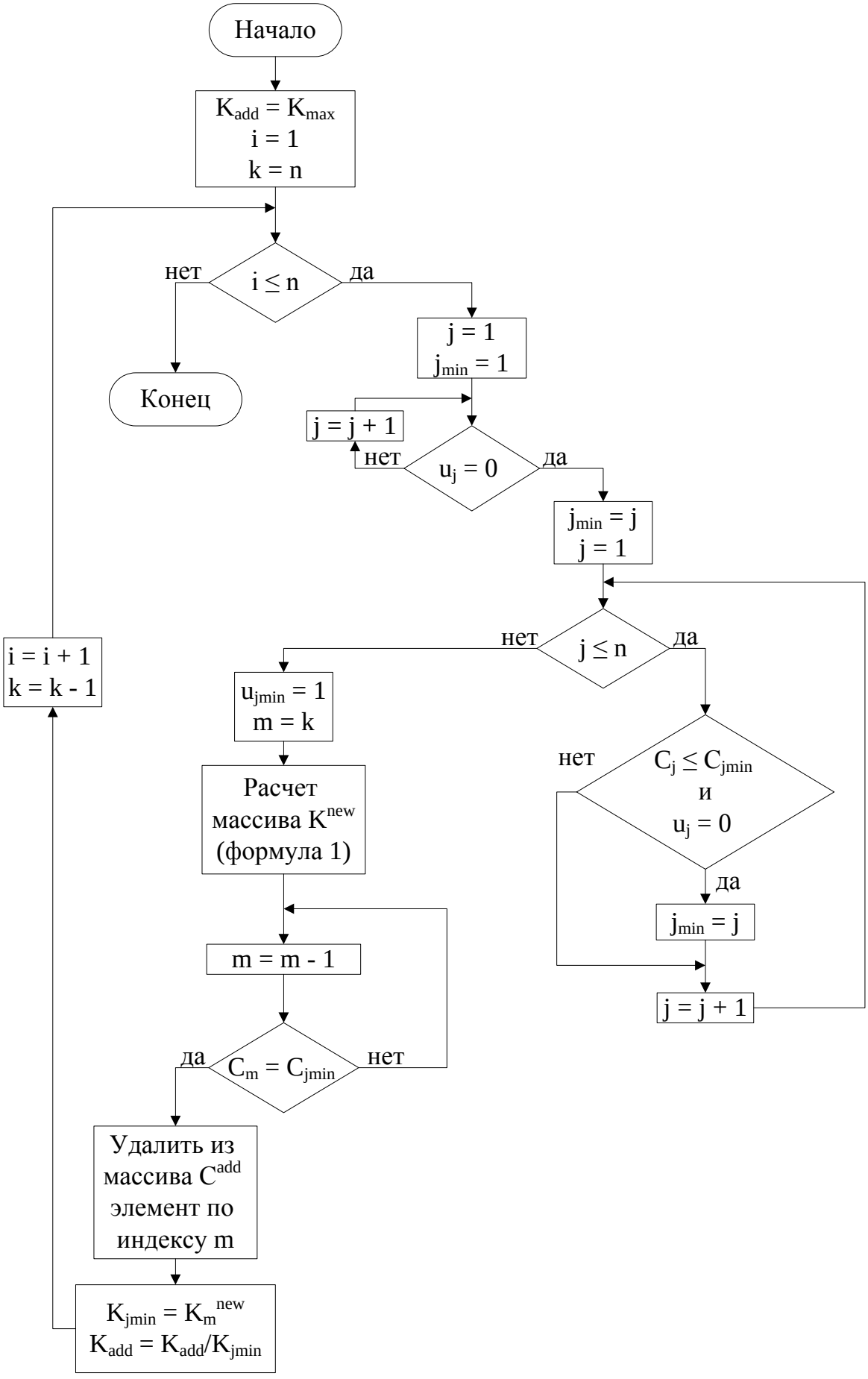

Рис. 2. Алгоритм определения количества вариаций

Расчет промежуточного количества тестов

$$t = \sqrt[k]{\frac{K_{add}}{\prod_{i=1}^{k} C_i^{add}}}; \qquad (1)$$

$$K_j^{new} = \left\lfloor C_j^{add} \cdot t \right\rfloor, j = 1..k,$$

где K_{add}, C_j^{add} – вспомогательные параметры, которые на начало тестирования равны K_{max} и C_j соответственно, $\lfloor x \rfloor$ – операция округления x до ближайшего меньшего целого.

Важно отметить, что на практике затраты могут быть определены не максимально возможным количеством тестов K_{max}, а максимальным временем тестирования t_{test}^{max}. Тогда для применения рассмотренного алгоритма необходимо найди взаимно-однозначное соответствие между K_{max} и t_{test}^{max}. Определим максимально возможное количество тестов по формуле:

$$K_{max} = \left\lfloor \frac{t_{test}^{max}}{t_{mod}^{R}} \right\rfloor,$$

где t_{mod}^{R} – среднее время работы модуля, которое можно определить воспользовавшись формулой математического ожидания дискретно заданной величины:

$$t_{mod}^{R} = t_{mod}^{max} \cdot (1 - R_{mod}) + t_{mod}^{100}(R_{mod}),$$

где R_{mod} – вероятность безотказной работы метода, t_{mod}^{100} – усредненное время работы модуля при $R_{mod} = 1$, t_{mod}^{max} – максимальное время работы модуля (задано пользователем). При времени работы модуля $t_{mod} \geq t_{mod}^{max}$ считается, что произошел сбой.

Кроме того, среднее время работы модуля может быть определено на основе оценки предыдущего тестирования модуля (в случае, если тестирование этого модуля уже проводилось раннее):

$$t_{mod}^{R} = \frac{t_{cur}}{K_{cur}}.$$

Построенная схема тестирования применима при условии, что найдены значения приоритетов C_i. Рассмотрим некоторые способы их определения.

Первый способ. Каждому разработчику алгоритма задачи РКПЗ известна её специфика, область применения и т.д. На основе опыта разработчик может вручную сопоставить каждому входному параметру его приоритет. Основной недостаток этого подхода – человеческий фактор.

Второй способ основан на определении показателя вычислительной устойчивости ПМ относительно вариаций каждого из входных параметров.

Построение приоритетов входных параметров

Пусть известно количество узлов в n-мерной сетке для каждого параметра K_i, максимально возможное значение приоритета C_{max}, необходимо определить приоритеты входных параметров C_i^{cur}.

Воспользуемся следующим алгоритмом:

1) Для каждого из входных параметров p_i построим $s = \frac{K_{cur}}{K_i}$ матриц $P_1 \dots P_s$ вида

$$\begin{pmatrix} p_1^1 & \cdots & p_i^1 & \cdots & p_n^1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ p_1^1 & \cdots & p_i^{K_i} & \cdots & p_n^1 \end{pmatrix}, \begin{pmatrix} p_1^1 & \cdots & p_i^1 & \cdots & p_n^2 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ p_1^1 & \cdots & p_i^{K_i} & \cdots & p_n^2 \end{pmatrix} \dots$$

$$\dots\begin{pmatrix} p_1^{K_1} & \cdots & p_i^1 & \cdots & p_n^{K_n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ p_1^{K_1} & \cdots & p_i^{K_i} & \cdots & p_n^{K_n} \end{pmatrix},$$

где $P_1 \dots P_s$– матрицы полученные, при фиксации значений параметра p_i и при вариациях всех остальных параметров.

2) Каждой из построенных матриц сопоставим матрицу выходных параметров $V_1 \dots V_s$:

$$\begin{pmatrix} v_{11}^1 & \cdots & v_{1k}^1 \\ \vdots & \ddots & \vdots \\ v_{m1}^1 & \cdots & v_{mk}^1 \end{pmatrix} \dots \begin{pmatrix} v_{11}^s & \cdots & v_{1k}^s \\ \vdots & \ddots & \vdots \\ v_{m1}^s & \cdots & v_{mk}^s \end{pmatrix}, m = K_i ,$$

$$\begin{pmatrix} v_{11}^i & \cdots & v_{1k}^i \\ \vdots & \ddots & \vdots \\ v_{m1}^i & \cdots & v_{mk}^i \end{pmatrix} \to \left(\sigma_1^i \dots \sigma_k^i\right) = \begin{pmatrix} \sigma_1^i \\ \vdots \\ \sigma_k^i \end{pmatrix}^T ,$$

где σ_j^i – среднеквадратичное отклонение j-того столбца, j=1..k.

3) Меняя индекс $i = 1..s$, сформируем вектор $\vec{q} = \{q_1, q_2, .., q_k\}$:

$$\begin{pmatrix} \sigma_1^1 & \cdots & \sigma_1^s \\ \vdots & \ddots & \vdots \\ \sigma_k^1 & \cdots & \sigma_k^s \end{pmatrix} \to \begin{pmatrix} q_1 \\ \vdots \\ q_k \end{pmatrix}, \text{где } q_i = \frac{\sum_{l=1}^{s} \sigma_i^l}{s \cdot \sqrt{\sum_{l=1}^{s} (\sigma_i^l)^2}}.$$

4) Определяем среднее арифметическое значение q_{sum}:

$$q_{sum} = \frac{\sum_{j=1}^{k} q_j}{k}.$$

Повторив представленный алгоритм для каждого из входных параметров, получим вектор $\overrightarrow{q_{\text{ср}}}$, i=1..n. Вектор приоритетов определим в виде

$$C_i^{\text{вх}} = \frac{q_{\text{ср}}^i}{\left|\overrightarrow{q_{\text{ср}}}\right|} \cdot C_{max} ,$$

$\left|\overrightarrow{q_{\text{ср}}}\right|$– длина вектора $\overrightarrow{q_{\text{ср}}}$.

Стоит отметить, что для построения приоритетов вторым способом, необходимо иметь результаты предыдущего тестирования ПМ. Первое же тестирование можно провести задав одинаковые приоритеты входных параметров.

Результат

Эффективность применения разработанной схемы построения расчетной сетки тестирования покажем в сравнении с равномерной расчетной сеткой (полученной при условии, что значение всех приоритетов $C_i^{cur} = 1$, i =1..n). В качестве критерия для сравнения возьмем количество найденных наборов исходных данных, приводящих к отказу в работе ПМ.

Тестирование проводилось на 7 ПМ в два этапа – на начальном этапе разработки ПМ и на этапе регрессионного тестирования. Для каждого ПМ составлено от 6 до 15 сценариев тестирования.

Список тестируемых программных модулей:

1) расчет пассивного участка траектории методом Кеплера [12];
2) расчет пассивного участка с преобразованием уравнений движения по методу Кустаанхеймо-Штифеля [13 –15];
3) расчет баллистических производных [16];
4) расчет пассивного участка методом Рунге-Кутта 4-го порядка [17];
5) прямая баллистическая задача [16, 18];
6) обратная баллистическая задача [16, 18];
7) расчет траектории движения летательного аппарата [16, 18-20].

Результат тестирования можно представить в виде графиков. По оси абсцисс расположены ПМ, по оси ординат – количество найденных отказов.

В результате тестирования были построены 2 графика. Первый график соответствует начальному этапу разработки ПМ (рис. 3). Второй график соответствует регрессионному тестированию – после анализа результатов первого тестирования и исправления ошибок (рис. 4). Регрессионное тестирование – тестирование, направленное на проверку того, что в ранее работоспособной программе (после модернизации или исправлений) не появилось новых ошибок.

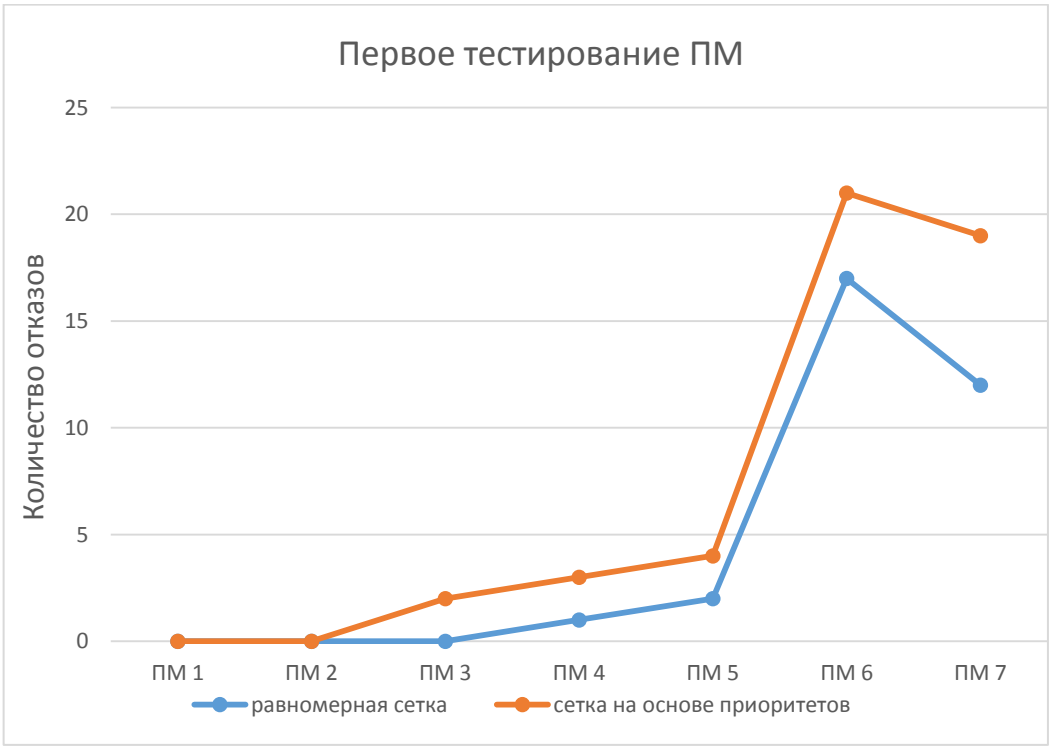

Рис. 3. Результат первого тестирования

Рис. 4. Результат регрессионного тестирования

Графики показывают, что тестирование по расчетной сетке, сформированной на основе построения приоритетов исходных данных (показано красной цветом), позволяет отследить большее количество отказов в сравнении в сравнении с тестированием по равномерной расчетной сетке (показано синим цветом). Таким образом, применение рассмотренного алгоритма, позволяет исправить большее количество ошибок на этапе отработки ПМ и тем самым обеспечить его безотказность и надежность.

Заключение

Разработанная схема проведения тестирования задач РКПЗ на основе показателей их вычислительной устойчивости позволяет уточнить область варьирования исходных данных и тем самым эффективнее использовать отведенное на этап тестирования время и получать более качественную оценку работоспособности ПО РКПЗ.

В качестве недостатка разработанной схемы можно отметить следующее. Алгоритм определения количества вариаций параметров (рис. 2) подразумевает возможность неограниченного разбиения диапазона изменения каждого из входных параметров. Однако, существуют целочисленные параметры, имеющие ограниченное число возможных вариаций (например, признак крутизны траектории, признак типа головного обтекателя). Этот недостаток можно устранить путем модернизации представленного алгоритма.

Разработанную схему тестирования можно использоваться не только для тестирования ПО РКПЗ, но для определения безотказности других программных функций (например, реализующих численные методы, рекурсию, итерационные процессы и т.д.).

Библиографический список

1. Ляпин А.А. Методология тестирования программного обеспечения расчета и контроля полетного задания // Материалы XX Юбилейной Международной конференции по вычислительной механике и современным прикладным программным системам (ВМСППС'2017), Алушта, 24-31 мая 2017. – М.: Изд-во МАИ, 2017. С. 154-156.

2. Ляпин А.А., Голунов М.С. Технология разработки программного обеспечения расчета и контроля полетного задания для межконтинентальных баллистических ракет // V Ежегодный форум «Информационные технологии на службе оборонно-промышленного комплекса – 2016». Челябинск, 17-20 мая 2016. – Челябинск: Издательский дом Connect. 2016. С. 62.

3. Тюгашев А.А., Ильин И.А., Ермаков И.Е. Пути повышения надежности и качества программного обеспечения в космической отрасли // Управление большими системами. 2012. № 39. С. 288 - 299.

4. James Bach. What is Exploratory Testing? URL: http://www.satisfice.com/articles/what_is_et.shtml

5. Савин Р. Тестирование Дот Ком, или Пособие по жестокому обращению с багами в интернет-стартапах. - М.: Дело, 2007. - 312 с.

6. Куликов С.С. Тестирование программного обеспечения. - Минск: Четыре четверти, 2015. – 294 с.

7. Дастин Э., Рэшка Д., Пол Д. Автоматизированное тестирование программного обеспечения: внедрение, управление и эксплуатация. - М.: ЛОРИ, 2003. - 567 с. ISBN 5-85582-186-2.

8. Бейзер Б. Тестирование черного ящика. Технологии функционального тестирования программного обеспечения и систем. – СПб.: Питер, 2004. - 318 с. ISBN 5-94723-698-2.

9. Котляров В.П., Коликова Т.В. Основы тестирования программного обеспечения - М.: БИНОМ. Лаборатория знаний, 2006. - 285 с.

10. Блэк Р. Ключевые процессы тестирования. – М: Лори, 2006. - 544с.

11. Николаев М.А., Юферов А.Г. Алгоритмы организации вариантных проектных расчетов // Научно-технический вестник Поволжья. 2013. № 6. С. 388 – 393.

12. Эскобал П. Методы определения орбит. - М: Мир, 1970. - 472 с.

13. Штифель Е., Шейфеле Г. Линейная и регулярная небесная механика. – М.: Наука, 1975. - 304с.

14. Иванюхин А.В. Оптимизация траектории космического аппарата с идеально регулируемым двигателем в переменных Кустаанхеймо-Штифеля // Труды МАИ. 2014. № 75. URL: http://trudymai.ru/published.php?ID=49691

15. Бордовицына Т.В., Авдюшев В.А. Теория движения искусственных спутников Земли. Аналитические и численные методы - Томск: Изд-во Томского университета, 2007. - 178 с.

16. Аппазов Р.Ф., Сытин О.Г. Методы проектирования траекторий носителей и спутников Земли. - М.: Наука, 1987. – 440 с.

17. Бордовицына Т.В. Современные численные методы в задачах небесной механики. - М.: Наука, 1984. – 136 с.

18. Сихарулидзе Ю.Г. Баллистика летательных аппаратов. - М.: Наука. 1982. - 352 с.

19. Косова А.С. Особенности отработки программно-алгоритмического обеспечения системы управления движением и формирования консервативной информации одного типа летательных аппаратов // Труды ФГУП НПЦАП. Системы и приборы управления. 2017. № 2. С. 27 - 36.

20. Соммервилл И. Инженерия программного обеспечения. – М.: Издательский дом «Вильямс», 2002. – 624 с.