УДК (004.4’236:004.424.2):629.78.064.018

Ю.А. КУЗНЕЦОВА, И.Б. ТУРКИН

Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Украина

ВИЗУАЛИЗАЦИЯ ТРАССЫ УПРАВЛЯЮЩЕГО АЛГОРИТМА В СИСТЕМАХ АВТОМАТИЗАЦИИ ИСПЫТАНИЙ АВИАЦИОННЫХ ДВИГАТЕЛЕЙ

Показано, что на современном этапе развития авиационной техники требуются новые подходы к разработке программного обеспечения, реализующего управляющий алгоритм испытаний авиационных двигателей, которые повысят наглядность программного обеспечения, обеспечат снижение трудоёмкости процессов тестирования и отладки управляющих алгоритмов, а также возможность их дальнейшей модификации и редактирования без привлечения высокооплачиваемых квалифицированных программистов. Предложена формальная модель трассы управляющего алгоритма, а также метод её визуализации, в основе которого лежит концепция недоопределенности и потоковых вычислений на моделях с недоопределенными значениями.

Ключевые слова: *авиационные двигатели, автоматизация испытаний, управляющий алгоритм реального времени, визуализация, графовые модели, программирование в ограничениях.*

Введение

Современные авиационные двигатели являются наиболее дорогостоящими, энергоёмкими и высоконагруженными элементами самолёта, которые для обеспечения высоких экономических показателей работают в условиях повышенных тепловых и силовых нагрузок, что требует особого внимания к обеспечению надёжности двигателя в полёте.

В конструкции современных двигателей предусмотрено наличие совершенных и развитых систем встроенного контроля, обнаружения и распознавания неисправностей в полёте и при техническом обслуживании. Эти системы собирают информацию о работе двигателей в полёте, регистрируют её и в случае необходимости выдают информацию о неисправностях на индикаторы мониторов пилотов и в виде распечаток на принтер.

В эксплуатационной практике наибольшее развитие находят системы, предусматривающие использование бортовых средств контроля и накопления информации о техническом состоянии двигателя, которые позволяют проводить оценку исправности, работоспособности, правильности функционирования и поиск неисправности до съёмного узла [1, 2]. Основными используемыми в настоящее время методами обеспечения качества и глубины диагностирования таких систем являются тестирование и отладка управляющих алгоритмов реального времени (УА РВ), которые не могут гарантировать отсутствия ошибок. При этом, в связи с необходимостью отработки взаимодействия с бортовой аппаратурой (БА) при всех возможных ситуациях на борту самолёта (в т.ч. нештатных), их трудоёмкость является наибольшей среди этапов жизненного цикла (ЖЦ) программного обеспечения (ПО) испытаний и составляет, по экспертным оценкам, около 57% общей трудоёмкости [3]. Учитывая сжатые сроки, необходимы более эффективные подходы и средства к разработке ПО, реализующего УА испытаний авиационных двигателей (АД).

Одним из таких подходов является визуализация УА ПО, наглядно представляющая оператору и разработчику летательного аппарата (самолёта) последовательность и содержание режимов и проверок, предусмотренных документацией, для определения оптимального (по продолжительности и глубине проверок) алгоритма испытаний АД.

1. Анализ исследований и публикаций

Анализ систем автоматизации испытаний АД. Одним из важных этапов процесса создания авиационных газотурбинных двигателей (ГТД) является проведение их испытаний на стендах опытно-конструкторского бюро, завода изготовителя, а также контроля в процессе эксплуатации. Автоматизация испытаний АД позволяют сократить время испытаний на 25…35%, снизить время обработки результатов в 2…2,5 раза, повысить информативность и объективность контроля на 25…30%, уменьшить численность обслуживающего персонала и экономить электроэнергию и топливо [4, 5].

Автоматизация испытаний является одним из важнейших факторов, позволяющих повысить качество проведения испытаний и достоверность получаемых результатов [6]. Кроме того, автоматизация позволяет сократить время на проведение испытаний и многократно увеличить производительность.

Применение автоматизированных систем контроля испытаний (АСКИ) наиболее целесообразно при контроле во время газовой наработки ГТД [7]. Это обусловлено, в первую очередь, тем, что:

1) именно в этих работах определяются значения основных показателей работы двигателя;

2) необходимо минимизировать газовую наработку (без ухудшения качества испытания) для снижения расхода топлива и для уменьшения выброса отходов горения топлива в атмосферу;

3) необходимо снизить влияние субъективного фактора для повышения объективности результатов испытания.

В отличие от автоматических систем автоматизированные системы в своем составе предполагают функционирование человека. Для эффективной работы людей все оперативные данные, характеризующие работу изделия, должны быть представлены в удобном наглядном виде [8].

Помимо эффективного представления оперативных данных также в удобном виде должны быть выведены значения, отправляемые в протокол испытания. Итоговое заключение о пригодности ГТД к эксплуатации принимается на основании данных, представленных в протоколе испытания [9]. Поэтому данные, заносимые в протокол, должны быть достоверны. Достоверность обеспечивается соответствием их обработки нормативной документации. Вместе с тем достоверность данных должна быть обеспечена требуемой скоростью ввода данных [10].

В цикле работы автоматизированной системы большое количество времени занимает визуализация данных. По этой причине достичь большой скорости работы ПО сложно. Это связано со скоростью реакции человека, составляющей в среднем 0,1 с. Поэтому увеличение скорости работы цикла отображения программы свыше 10 Гц нецелесообразно из-за того, что информация, выводимая на устройства визуализации, не сможет быть воспринята пользователями. С другой стороны, чтобы обеспечить требуемую точность и достоверность отдельных параметров, необходимо иметь скорость цикла работы не ниже 130 Гц [11, 12].

Для решения этой проблемы АСКИ разбивается на две крупных подсистемы – систему сбора данных и систему визуализации. Система сбора данных работает в реальном режиме времени технологического процесса, а система визуализации – в реальном времени реакции человека, которое на порядки больше дискретности, которую необходимо поддерживать, чтобы эффективно контролировать испытание [13]. Разделение системы контроля на подсистемы сбора данных и визуализации позволяет регистрировать параметры ГТД с высокой точностью и эффективностью.

Визуализация УА. *Управляющий алгоритм* – алгоритм верхнего уровня, управляющий в реальном времени виртуальной машиной, в которой реализованы элементарные операции ввода/вывода и преобразования информации, а также выдачи в каждый заданный момент времени t_i (или на протяжении заданного промежутка времени $\Delta\tau$) корректного управляющего воздействия.

Известно достаточно много графических представлений управляющих алгоритмов, таких как блок-схемы, временные диаграммы, диаграммы состояний и переходов, деятельности и т.д. Тем не менее, *универсальным средством представления управляющих алгоритмов являются графы* [14]. Поэтому методы визуализации графовых структур представляют собой теоретическую основу методов визуализации управляющих алгоритмов.

Чтобы повысить наглядность изображения графа принято руководствоваться *эстетическими критериями*: минимизация количества пересечений и сгибов рёбер; минимизация количества наложений рёбер и вершин и области размещения; максимизация разрешения; минимизация общей длины рёбер; минимизация длины ребра; унификация длин рёбер; минимизация сгибов на ребре; унификация сгибов рёбер; максимизация симметричности изображения; минимизация коэффициента сторон.

Проблемой является то, что невозможно удовлетворить все критерии одновременно из-за их взаимопротиворечивости или из-за сложности алгоритмической реализации. Имеется ряд методов, которые удовлетворяют только отдельным эстетическим критериям в задачах визуализации графов: *планаризация, использование физических аналогий, Сигаяма-подобные методы, потоковые методы*. Но они не являются универсальными и используются только для рисования статических графов. Поэтому возникает необходимость применения методов программирования в ограничениях, позволяющих строить интерактивные изображения графов.

2. Постановка задачи

Из-за сложности и иерархичности УА, изменения последовательности, объёма и режимов проверок становится затруднительным описание правил представления алгоритмов сбора и преобразований данных, формирования управляющих воздействий и выдачи сообщений оператору.

Поэтому *целью* данной работы является повышение наглядности программного обеспечения, реализующего управляющий алгоритм в системах автоматизации испытаний авиационных двигателей.

При этом под *наглядностью* (от англ. affordance) в соответствии с ISO DIS 9241 [15] подразу-

мевается наиболее значимая составляющая «понятного интерфейса». Единственным способом передачи такого свойства является изображение, т.е. мнемосхема процесса испытаний, основанная на метафоре представления этого процесса в достаточно простом и удобном для восприятия оператором виде. Польза наглядности заключается в том, что она позволяет пользователям обходиться без какого-либо предварительного обучения, и благодаря этому является самым эффективным и надежным средством обеспечения понятности.

Для достижения поставленной цели необходимо разработать модель трассы управляющего алгоритма и метод её визуализации.

3. Формальная модель трассы управляющего алгоритма реального времени

Пусть обработка информации в ПО испытаний АД производится в соответствии с **моделью вычислений, управляемых данными** [16]:

$$\text{МВ\_УД} = <\text{Data, Task, DT, TD}>, \quad (1)$$

где $\text{Data} = \{\text{Data}_i\}$ – множество объектов данных, $i = \overline{1,N}$; $\text{Task} = \{\text{Task}_j\}$ – множество элементарных задач-преобразователей информации, $j = \overline{1,M}$; $\text{DT} = \text{Data} \times \text{Task}$ – матрица, характеризующая правила инициирования задач при изменении (обновлении) данных; $\text{TD} = \text{Task} \times \text{Data}$ – матрица, характеризующая получение (порождение) новых данных в результате работы задач-преобразователей информации.

Данные. Формально объект данных $\text{Data}_i \in \text{Data}$ можно описать с помощью кортежа:

$$\text{Data}_i = \langle \text{DataValue}_i,\ \text{TimeStamp}_i,\ \text{Quality}_i \rangle,$$

где DataValue_i – значение i-того объекта данных в некоторый TimeStamp_i момент времени (отметка времени), Quality_i – качество i-того объекта данных, $i = \overline{1,N}$ [17].

Задачи. В результате работы преобразователей информации $\text{Task}_i \in \text{Tasks}$ будут порождаться объекты данных – $\text{Data}_{\text{out}_{\text{Task}_i}}$. Поэтому целесообразно рассматривать расширенное множество объектов данных как объединение двух непересекающихся подмножеств данных: сенсорных – $\text{Data}_{\text{sensor}_{\text{Task}}}$, полученных из внешнего мира, и вычисляемых – $\text{Data}_{\text{calculated}}$, то есть порождаемых работой задач Tasks.

Правила инициирования задач. Пусть $e \in \text{DT}$. Ребро **e** означает наличие связи, при этом вес ребра, характеризующий тип этой связи:

$$w_e = \begin{cases} 1 - \text{данные используются задачей } \text{Task}_j; \\ 2 - \text{по изменению } \text{Data}_i \text{ инициируется } \text{Task}_j; \\ 3 - \text{по обновлению } \text{Data}_i \text{ инициируется } \text{Task}_j; \\ i = \overline{1,N},\ j = \overline{1,M}. \end{cases}$$

Следовательно, УА представляет собой множество подграфов G_{DataCA_i} общего графа вычислений G_{Data} МВ_УД, которые формируют выходные данные $\text{Data}_{\text{out}_{\text{Task}_i}}$ как команды управления Y:

$$\begin{gathered} \text{CA} = \left\{ G_{\text{DataCA}_i} \right\},\ G_{\text{DataCA}_i} \subset G_{\text{Data}}: \\ \forall G_{\text{DataCA}_i} \in G_{\text{Data}}, \\ G_{\text{DataCA}_i} : \left(\text{Data}_{\text{out}_{\text{Task}_i}} \right) \longrightarrow \text{Y}. \end{gathered} \quad (2)$$

Следовательно, УА РВ целесообразно представить в виде кортежа:

$$\text{CA RT} = \left\langle \begin{matrix} \text{Data}_{\text{CA}},\ \text{Task}_{\text{CA}}, \\ \text{Data}_{\text{CA}} \times \text{Task}_{\text{CA}}, \\ \text{Task}_{\text{CA}} \times \text{Data}_{\text{CA}} \end{matrix} \right\rangle, \quad (3)$$

где $\text{Data}_{\text{CA}} \subset \text{Data}$; $\text{Task}_{\text{CA}} \subset \text{Task}$; $\text{Data}_{\text{CA}} \times \text{Task}_{\text{CA}} \subset \text{DT}$; $\text{Task}_{\text{CA}} \times \text{Data}_{\text{CA}} \subset \text{TD}$.

Пусть известно множество команд, формируемых УА: $\text{Data}_{\text{КУ}} \subset \text{Data}_{\text{CA}}$. Тогда УА как множество данных, задач и связей между ними (3) можно определить рекурсивно:

1) определяется множество задач 0-го уровня $\text{Task}^0 = \{t_j\}$, $j = \overline{1,M}$ таким образом, что:

$$\begin{gathered} \forall t_j \in \text{Task}^0, \exists d_i \in \text{Data}^0, \\ (\text{Data}^0 \equiv \text{Data}_{\text{КУ}}) \wedge (td_{ji} \in \text{TD}); \end{gathered} \quad (4)$$

2) определяется множество данных 1-го уровня $\text{Data}^1 = \{d_i\}$, $i = \overline{1,N}$ таким образом, что:

$$\forall d_i \in \text{Data}^1, \exists t_j \in \text{Task}^0,\ dt_{ij} \in \text{DT}; \quad (5)$$

3) множество задач k-го уровня $(k \geq 1)$ $\text{Task}^{(k)}(t_j)$ определяется по известному множеству данных k-го уровня $\text{Data}^{(k)}(d_i)$ и известному множеству $\text{TD}(td_{ji})$:

$$\forall t_j \in \text{Task}^k, \exists d_i \in \text{Data}^k,\ td_{ji} \in \text{TD}; \quad (6)$$

4) множество данных k-го уровня $(k \geq 1)$ $\text{Data}^{(k)}(d_i)$ определяется по известному множеству задач $\text{Task}^{(k-1)}(t_j)$ (k-1)-уровня и известному множеству $\text{DT}(dt_{i-1,j})$:

$$\forall d_i \in \text{Data}^k, \exists t_j \in \text{Task}^{k-1},\ dt_{i\text{-}1,j} \in \text{DT}. \quad (7)$$

Вычисления (4) – (7) останавливаются, когда выполняется условие:

$$((\forall t_j \in Task^k) \wedge (\forall d_i \in Data^k)) \exists \\ \exists((dt_{ij} \in DT) \wedge (d_i \in Data_{in})), \quad (8)$$

где $Data_{in} = \{Data_{in}\}$;

$Data_{in} \subset \left\{Data^k\right\}$, $k = 0\ldots(n-1)$.

Формулы (4) – (8) означают, что были последовательно по всем (n-1) уровням определены все объекты данных d_i УА и все задачи-преобразователи t_j этих данных, при этом $i = \overline{1,N}$; $j = \overline{1,M}$; $k = 0\ldots(n-1)$ – уровень УА; $Data_{in} \equiv Data_{sensor_{Task}}$.

Для того, чтобы упорядочить полученные при *обратном проходе* УА вершины Data и Task графа G_{DataCA_i} по $(n-1)$ уровням, необходимо выполнить сортировку этого графа при *прямом проходе* УА:

$$^*Data^{(n-1)} = Data^{(n-1)} - Data_{in}; \quad (9)$$

$$^*Task^{(n-2)} = Task^{(n-2)} - Task^{(n-1)}; \quad (10)$$

$$^*Data^{(k)} = Data^{(k)} - \bigcup_{i=k-1}^{n-1} Data^{(i)},\ k = (n-2)\ldots1; \quad (11)$$

$$^*Task^{(k)} = Task^{(k)} - \bigcup_{i=k-1}^{n-1} Task^{(i)},\ k = (n-2)\ldots1. \quad (12)$$

Правила формирования контрольных точек (КТ). Из-за различий временной организации управляющей программы и программной модели УА возникает проблема установления соответствия при сравнении результатов обеих программ. Исходным решением является назначение КТ в УА.

Контрольная точка – это точка в адресно-временном пространстве программы, представленная в общем случае всеми параметрами текущего состояния программы. КТ идентифицируется уникальным именем и порядковым номером [18].

Для обеспечения синхронизации данных при отработке необходимо связывание КТ с событиями и параметрами УА, а не с РВ непосредственно. Это решает проблему установления соответствия данных при сравнении фактических и эталонных процессов при работе управляющей программы во время её отработки. С КТ связываются технологические операции. Эта связь задается описанием КТ, содержащим перечень операций, которые необходимо выполнить в моменты её реализации. Описание контрольной точки имеет следующий формат:

$$PMP = \langle ID, Uname, oper\_type_k, Pr_k, Data_k, Task_k \rangle, \\ k = \overline{1,K},$$

где ID – идентификационный (порядковый) номер контрольной точки; UName – уникальное имя КТ; $oper\_type_k$ – тип выполняемой технологической операции; Pr_k – условия записи контрольных точек, $Data_k \subset Data$, $Task_k \subset Task$.

Условия записи контрольных точек задаются множеством предикатов Pr по состоянию данных pr_{Data_m} и задач pr_{Task_m}, обуславливающих выполнение той или иной задачи $Task_m$ в зависимости от набора входных данных Data для этой задачи:

$$Pr_m = \left\{\left\langle pr_{Data_m}, pr_{Task_m}, W_{Data_m}, W_{Task_m}\right\rangle\right\}.$$

Pr_m – сложный предикат, определённый на множестве Task и Data, при этом его область определения $D_{Pr_k} = (-\infty; +\infty)$, а область значений $E_{Pr_k} = \{0,1\}$: 0 – значение $Data_m$ не изменилось, 1 – обновление (изменение) информации о состоянии Data и Task; $Data_m \in Data$, $Task_m \in Task$.

W_{Data_m} – КТ по текущему состоянию требует записи объекта данных $Data_m$.

W_{Task_m} – КТ в результате преобразования требует записи задачи $Task_m$.

m – список индексов (записей, имён), $m = \overline{1,M}$.

Таким образом, **трасса управляющего алгоритма** (ТУА) представляет собой упорядоченное во времени τ множеством контрольных точек, полученное в процессе реализации УА РВ:

$$RTT = \left\langle \left\langle \tau, PMP_i \right\rangle_j, j \in 1\ldots M \right\rangle, i \in 1\ldots N.$$

Для её визуализации разрабатывается метод, основанный на решении задачи распространения ограничений на изображение графа.

4. Представление задачи визуализации трассы управляющего алгоритма в виде задачи удовлетворения ограничений

Повышение эффективности визуального представления ТУА достигается за счёт решения **оптимизационной задачи распространения ограничений на изображение графа**, которая сводится к нахождению экстремального значения целевой функции на основе полученных исходных данных, удовлетворяющих заданным ограничениям.

Исходные данные (inputs):

– длина ScreenWidth и высота ScreenHeigth монитора;

– координаты центра, левого верхнего и левого нижнего углов, правого верхнего и правого нижнего углов;

– множество объектов данных Data;

– множество элементарных задач-преобразователей информации Task;

– матрица DT, характеризующая правила инициирования задач при изменении (обновлении) данных;

– матрица TD, характеризующая получение (порождение) новых данных в результате работы задач-преобразователей информации.

Ограничения:

1) только положительные координаты: $x_i>0$, $y_j>0$;

2) ограничения на размер вершины $Data_i$:

$Size_{vert_{max}} = ScreenHeigth$ – максимальное значение высоты вершины $Data_i$;

$Size_{vert_{min}} = const$ – минимальное значение высоты вершины $Data_i$;

$Size_{hor_{max}} = ScreenWidth$ – максимальное значение длины вершины $Data_i$;

$Size_{hor_{min}} = const$ – минимальное значение длины вершины $Data_i$;

$$Size_{vert_{min}} \le Size_{vert} \le Size_{vert_{max}};$$

$$Size_{hor_{min}} \le Size_{hor} \le Size_{hor_{max}};$$

3) ограничения на размер вершины $Task_j$:

$R_{max} = const$ – максимальное значение радиуса вершины $Task_j$;

$R_{min} = const$ – минимальное значение радиуса вершины $Task_j$;

$$R_{min} \le R \le R_{max};$$

4) ограничения на толщину линий TD и DT, $Data_i$ и $Task_j$:

$line\_thick_{max} = const$ – максимальное значение толщины линий TD и DT, $Data_i$ и $Task_j$;

$line\_thick_{min} = const$ – минимальное значение толщины линий TD и DT, $Data_i$ и $Task_j$;

$$line\_thick_{min} \le line\_thick \le line\_thick_{max};$$

5) ограничения на длину направленных рёбер:

$line\_length_{max} = const$ – максимальное значение длины направленных рёбер;

$line\_length_{min} = const$ – минимальное значение длины направленных рёбер;

$$line\_length_{min} \le line\_length \le line\_length_{max};$$

6) ограничение на совместное использование нескольких эстетических критериев (выполнение условий непротиворечивости).

Переменные:

– длина $Size_{hor}$ и высота $Size_{vert}$ вершины $Data_i$;

– радиус R вершины $Task_j$;

– толщина $line\_thick$ и длина $line\_length$ линий;

– эстетический критерий.

Целевая функция:

$$object\_func(inputs, variables).$$

Метод решения задачи распространения ограничений на изображение графа. Способ представления недоопределенного значения влияет как на качество полученных результатов, так и на вид ограничений, связывающих это значение. В зависимости от характера представляемой информации недоопределенные значения могут быть представлены в виде целочисленных и вещественных интервалов, множеств, перечислений и других, более специальных, конструкций [19].

Определение 1. *Задачей удовлетворения ограничений (ЗУО) называется пара (V,C), где V — совокупность переменных, принимающих значения из некоторых универсальных множеств (универсумов), а C – совокупность ограничений, связывающих значения переменных из V.*

Каждое ограничение из *C* имеет вид одной из следующих формул:

$$y = x;$$

$$y = c;$$

$$V = f\{x_1, ..., x_n\},$$

где x, x_i, y – символы переменных из V; c – константный символ; f – функциональный символ арности n.

Более сложные ограничения распадаются на множество более простых после введения дополнительных переменных [20].

Определение 2. *Решением задачи удовлетворения ограничений (V, C) называется такое приписывание каждой переменной из V некоторого определенного значения из её универсума, при котором выполняются все ограничения из C.*

В общем случае все точные решения задачи, если таковые существуют, должны лежать в декартовом произведении таких н-значений.

Определение 3. Для вычисления Н-модели *обобщённая вычислительная модель (н-модель) M* состоит из четырёх множеств:

$$M = (V, C, W, CORR),$$

где V – множество н-объектов v из заданной предметной области; C – множество ограничений на н-объектах из V; W – множество функций присваивания; CORR – множество функций проверки корректности.

С каждым объектом из V связаны недоопределенный тип данных (н-расширение некоторого универсума), начальное значение, функция присваивания и функция проверки корректности.

Каждая вершина $Data_i$ графа G_{DataCA_i} является структурным расширением интервального н-объекта, включая начальные $(x_0; y_0)$ и конечные $(x_n; y_n)$ координаты по осям абсцисс и ординат, а также высоту $Size_{vert}$ и длину $Size_{hor}$ и толщину линий квадрата $line\_thick$.

Каждая вершина $Task_j$ графа G_{DataCA_i} является структурным расширением интервального н-объекта, включая начальные $(x_0; y_0)$ и конечные $(x_n; y_n)$ координаты по осям абсцисс и ординат, а также радиус *R* и толщину окружности $line\_thick$.

Каждое направленное ребро DT и TD графа G_{DataCA_i} является структурным расширением интервального н-объекта, включая начальные $(x_0; y_0)$ и конечные $(x_n; y_n)$ координаты по осям абсцисс и ординат, а также длину $line\_length$ и толщину $line\_thick$.

Отношения непосредственного предшествования между вершинами $Data_i$ и $Task_j$, характеризущие частичную упорядоченность этих вершин, задаются в виде неравенств и входят в общую систему *C* – множества ограничений на н-объектах из *V*. В начале вычислений для всех вершин начальные и конечные координаты совпадают с координатами нижнего левого угла экрана монитора, а в результате вычислений происходит уточнение этих координат.

Координаты двух типов вершин $Data_i$ и $Task_j$, их размеры (высота $Size_{vert}$ и длина $Size_{hor}$, толщина линий квадрата $line\_thick$ вершины $Data_i$; радиус R и толщина окружности $line\_thick$ вершины $Task_j$), а также длина $line\_length$ и толщина $line\_thick$ направленных рёбер DT и TD определены как множество вещественных переменных. А их минимальные и максимальные значения представлены целыми константами. Совместное использование нескольких эстетических критериев определено во множестве констант логического типа.

Таким образом, исходная постановка оптимизационной задачи при использовании н-моделей заменена на поиск и определение такого множества значений параметров графа, при котором все показатели эффективности изображения графа удовлетворяют соответствующим ограничениям.

Заключение

Разработанная *модель трассы управляющего алгоритма реального времени* основана на совместном использовании модели вычислений, управляемых данными, и формальном описании контрольных точек, что обеспечивает высокую избирательность и глубину проникновения в отрабатываемую программу, учёт её логической структуры и последовательности протекающих процессов и событий.

Метод визуализации трассы управляющего алгоритма (ТУА) основан на предложенной Нариньяни А.С. концепции недоопределенности и потоковых вычислений на моделях с недоопределенными значениями. На основе данного метода сформулирована постановка задачи визуализации ТУА в виде задачи удовлетворения ограничений.

Перспективами настоящей работы является алгоритмизация методов распространения ограничений в задачах визуализации трассы управляющего алгоритма, а также разработка рабочего прототипа программного обеспечения для её визуализации.

Литература

1. Богуслаев, А.В. Информационные технологии поддержки жизненного цикла изделий в авиадвигателестроении [Текст] / А.В. Богуслаев, В.И. Дубровин, И.А. Набока // Радіоелектроніка. Інформатика. Управління. – 2004. – № 1. – С. 136–145.

2. Виноградов, Ю.В. Система экспресс-диагностики технического состояния авиационных ГТД [Текст] / Ю.В. Виноградов, А.П. Тунаков // Изв. вузов. Авиац. техника. – 2002. – № 1. – С. 78–79.

3. Тюгашев, А.А. Технология проектирования надёжных управляющих алгоритмов реального времени для космических аппаратов [Текст] / А.А. Тюгашев // Вестн. СГАУ. – 2004. – № 1. – С. 124–131.

4. Испытания воздушно-реактивных двигателей [Текст]: учеб. пособие / А.Я. Черкез и др. – М.: Машиностроение, 1992. – 304 с.

5. Дубровин, В.И. Интеллектуальные средства управления качеством: автоматизированная система «Диагностика» [Текст] / В.И. Дубровин, С.А. Субботин, А.В. Богуслаев // Вісник двигунобудування. – 2003. – № 1. – С. 156–161.

6. Звонарёв, С. Стендовый комплекс диагностики авиационных двигателей [Текст] / С. Звонарёв, В. Поклад, А. Потапов // Разработки. Авиация. – СТА, 2002. – С. 42–47.

7. Зеленков, Ю.А. Комплексная автоматизация испытаний газотурбинных двигателей. Ч. 1: управление стендом и сбор данных [Текст] / Ю.А. Зеленков, В.Ю. Чувилин В. Е. Журавлев // Вестн. УГАТУ. – 2011. – Т. 15, № 2 (42). – С. 119–125.

8. Измерительные приборы и системы [Электронный ресурс]. – Режим доступа к ресурсу: http://www.nppmera.ru. – 16.05.2012 г.

9. Солохин, Э.Л. Испытания авиационных воздушно-реактивных двигателей [Текст] / Э.Л. Солохин // Учеб. пособие по спец-ти «Авиационные двигатели». – М: Машиностроение, 1975. – 356 с.

10. Прогрессивные технологии моделирования, оптимизации и интеллектуальной автоматизации этапов жизненного цикла авиационных двигателей: моногр. [Текст] / А.В. Богуслаев, Ал.А. Олейник, Ан.А. Олейник, Д.В. Павленко, С.А. Субботин. – Запорожье: ОАО «Мотор Сич», 2009. – 468 с.

11. Автоматизация процесса испытания авиационных ГТД на базе SCADA-системы LabView [Текст] / Д.А. Ахмедзянов, Р.Р. Ямалиев, А.Е. Кишалов, А.В. Суханов // Вестн. УГАТУ. – 2009. – Т.13, №2 (35). – С. 61–68.

12. Интеллектуальные средства диагностики и прогнозирования надёжности авиадвигателей: Монография [Текст] / В.И. Дубровин, С.А. Субботин, А.В. Богуслаев, В.К. Яценко. – Запорожье: ОАО «Мотор Сич», 2003. – 279 с.

13. Испытания авиационных двигателей: учеб. пособие [Текст] / В.А Григорьев, С.П. Кузнецов, А.С. Гишваров, А. Н. Белоусов и др. – М: Машиностроение, 2009. – 504 с.

14. Касьянов, В.Н. Графы в программировании: обработка, визуализация и применение [Текст] / В.Н. Касьянов, В.А. Евстигнеев. – СПб.: БХВ-Петербург, 2003. – 1104 с.

15. ISO. ISO Standards [Электронный ресурс]. – Режим доступа к ресурсу: http://www.iso.org/iso/iso_catalogue.htm. – 16.05.2012 г.

16. Туркин, И.Б. Модель вычислений, управляемых данными, в программном обеспечении систем реального времени [Текст] / И.Б. Туркин, Е.В. Соколова // Радиоэлектронные и компьютерные системы. – 2010. – № 6 (47). – С. 13–18.

17. OPC Data Access Custom Interface: Industry Standard Specification, Version 3.00, Released [Text] / OPC Foundation. – March 4, 2003. – 190 p.

18. Теоретические основы проектирования информационно-управляющих систем космических аппаратов [Текст] / В.В. Кульба, Е.А. Микрин, Б.В. Павлов и др. – М: Наука, 2006. – 579 с.

19. Телерман, В.В. Удовлетворение ограничений в задачах математического программирования [Текст] / В.В. Телерман, Д.М. Ушаков // Вычислительные технологии. – 1998. – Т. 3, № 2. – С. 45–54.

20. Телерман, В.В. Недоопределенные модели: формализация подхода и перспективы развития [Текст] / В.В. Телерман, Д.М. Ушаков // Проблемы преставления и обработки не полностью определенных знаний. – М., 1996. – С. 7–32.

Поступила в редакцию 29.05.2012

Рецензент: д-р техн. наук, проф., зав. каф. информатики А.Ю. Соколов, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина.

ВІЗУАЛІЗАЦІЯ ТРАСИ КЕРУЮЧОГО АЛГОРИТМУ В СИСТЕМАХ АВТОМАТИЗАЦІЇ ВИПРОБУВАНЬ АВІАЦІЙНИХ ДВИГУНІВ

Ю.А. Кузнецова, І.Б. Туркін

Показано, що на сучасному етапі розвитку авіаційної техніки потрібні нові підходи до розробки програмного забезпечення, що реалізує керуючий алгоритм випробувань авіаційних двигунів, які підвищать наочність програмного забезпечення, забезпечать зниження трудомісткості процесів тестування та налагодження керуючих алгоритмів, а також можливість їх подальшої модифікації та редагування без залучення високооплачуваних кваліфікованих програмістів. Запропоновано формальну модель траси керуючого алгоритму, а також метод її візуалізації, в основі якого лежить концепція недовизначених і потокових обчислень на моделях з недовизначеними значеннями.

Ключові слова: авіаційні двигуни, автоматизація випробувань, керуючий алгоритм реального часу, візуалізація, графові моделі, програмування в обмеженнях.

THE VISUALIZATION OF CONTROLLING ALGORITHM TRACK IN THE SYSTEMS OF AUTOMATION TESTS OF AIRCRAFT ENGINES

Yu.A. Kuznetsova, I.B. Turkin

It is shown that at the present stage of aviation technology development requires the new approaches to the software developing that implements the test controlling algorithm of aircraft engines, which will increase the visibility of the software, provide a laboriousness reduction of test process and controlling algorithms debugging, as well as the possibility of its' sequel modifying and editing without the involvement of highly-paid and skilled programmers. A formal model of the controlling algorithm track, as well as the method of it's visualization, which is based on the concept of underdetermined and flow calculations on models with underdetermined values, are suggested in the paper.

Key words: aircraft engines, automation of tests, real-time controlling algorithm, visualization, graph models, constraint programming.

Кузнецова Юлия Анатольевна – аспирант каф. Инженерии программного обеспечения, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина, e-mail: JK-Sv@yandex.ru.

Туркин Игорь Борисович – доктор технических наук, профессор, зав. каф. Инженерии программного обеспечения, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков, Украина, e-mail: energy@d4.khai.edu.