УДК 004.75

ПРОБЛЕМЫ ПРОГРАММИРУЕМОСТИ, БЕЗОПАСНОСТИ И НАДЕЖНОСТИ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ И СЕТЕЦЕНТРИЧЕСКОГО УПРАВЛЕНИЯ
Ч. 2. Подход к общему решению

Ю.С. Затуливетер, Е.А. Фищенко

Изложены принципы формирования в глобальной компьютерной среде математически однородного, бесшовно программируемого и кибербезопасного алгоритмического пространства распределенных вычислений и сетецентрического управления на базе компьютерного исчисления древовидных структур. Приведены свойства языка бесшовного программирования в этом пространстве, а также принципы организации программными методами надежных распределенных вычислений в вычислительных средах с ненадежными компонентами. Сформулированы требования к сетевым компьютерам с немикропроцессорной архитектурой, которые необходимы для эффективной и кибербезопасной реализации данного алгоритмического пространства в глобальной компьютерной среде.

Ключевые слова: глобальная компьютерная среда, глобальная информационная сильносвязность, распределенные вычисления, сетецентрическое управление, исчисление древовидных структур, немикропроцессорная архитектура, математически однородное алгоритмическое пространство, бесшовное программирование, кибербезопасность, надежные вычисления в среде с ненадежными компонентами.

ВВЕДЕНИЕ

Количество компьютерных устройств разных классов (от «умных» сенсоров и смартфонов, до суперкомпьютеров), связанных глобальными сетями, исчисляется многими миллиардами и быстро растет. Совокупные вычислительные, функциональные и информационные ресурсы глобальной компьютерной среды (ГКС) обладают колоссальным системообразующим потенциалом. Для полномасштабного его раскрытия требуется массовое применение больших и сверхбольших систем распределенных вычислений, нацеленных на решение практически неограниченного разнообразия задач сетецентрического управления (СЦУ).

В первой части [1] настоящей работы показано, что с увеличением размеров и масштабов применения таких систем в условиях крайней разнородности аппаратных и программных платформ в ГКС возникают фундаментальные барьеры комбинаторной сложности задач функциональной интеграции вычислительных и информационных ресурсов. Существующие технологии построения распределенных вычислительных систем требуют лобового преодоления комбинаторной сложности задач интеграции. При этом с увеличением размеров, помимо неограниченного роста средств и времени, требуется добавление на системных уровнях очередных разнородных, все более сложных и менее надежных слоев промежуточного программного обеспечения (ПО).

В настоящее время доля вычислительных ресурсов, вовлекаемых в сильносвязные системы алгоритмической переработки глобально распределенной информации, относительно совокупных ресурсов ГКС скорее уменьшается, чем растет. Это говорит, прежде всего, о том, что общесистемная сложность ГКС, препятствующая созданию таких систем, достигла критических уровней, а также об отсутствии адекватного инструментария преодоления этой сложности.

В долгосрочной перспективе качественное развитие ГКС на основе существующих технологий интеграции, толерантных к разнородности, становится практически невозможным — ни в части

функциональной интеграции, ни в части обеспечения ее кибербезопасности и надежности.

В условиях растущих масштабов острых социальных проявлений кризиса перепроизводства в ГКС слабо формализованной информации [1] необходимы новые научно обоснованные подходы к общему решению проблем функциональной интеграции при создании больших систем распределенной переработки глобально распределенной информации в сколь угодно больших сетях.

В данной работе, в отличие от существующих технологических подходов к функциональной интеграции сетевых ресурсов при изначальной «легализации» их разнородности, рассматривается общий системный подход, цель которого состоит в устранении причин разнородности и бесшовном распространении свойства универсальной программируемости, присущего отдельному компьютеру, на сколь угодно большие множества компьютеров, связанных сетями.

Суть рассматриваемого подхода состоит в:

- выявлении и устранении первопричин непрерывного воспроизводства разнородных форм представления компьютерной информации (программ и данных) и способов работы с ней;
- разработке принципов и методов формирования единого, бесшовно программируемого и кибербезопасного алгоритмического пространства распределенных вычислений в сколь угодно больших сетях;
- разработке и обосновании требований к аппаратной поддержке этого пространства.

1. ПРИНЦИПЫ ФОРМИРОВАНИЯ МАТЕМАТИЧЕСКИ ОДНОРОДНОГО АЛГОРИТМИЧЕСКОГО ПРОСТРАНСТВА РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ

Чрезмерное разнообразие разнородных, трудно совместимых аппаратных платформ и форм представления данных и программ с позиций алгоритмической универсальности представляется заведомо избыточным. Оно приводит к многовариантности задач системно-функциональной интеграции сетевых ресурсов и к непреодолимой комбинаторной сложности создания и интеграции сколь угодно больших систем распределенной обработки данных в ГКС [1].

Устранение причин воспроизводства разнородности ГКС и проблем комбинаторной сложности интеграции разнородных ресурсов становится возможным посредством формирования в ней математически однородного алгоритмического пространства распределенных вычислений.

1.1. Ограничения классической компьютерной модели

Установлено [2—4], что первопричины разнородности информационных и вычислительных ресурсов компьютерных сред обусловлены свойствами так называемой модели Дж. фон Неймана [5], которая рассматривается как классическая индустриально значимая аксиоматика универсальных машинных вычислений [4]. Микропроцессорные компьютерные архитектуры стали ее массовым воплощением, базовым элементом компьютерной революции и глобализации сетевого пространства. По сути, из-за отсутствия индустриально значимых альтернатив, она до сих пор остается единой логической основой массового производства компьютеров и программ.

Сложившиеся к настоящему времени в ГКС условия и массовые проявления глобальной сильносвязности показывают принципиальные недостатки классической модели, из-за которых сбалансированное развитие ГКС становится невозможным. В этом состоит новизна и актуальность проблем и общего подхода к их решению, рассматриваемых в данной работе.

Классическая компьютерная аксиоматика изначально представляет собой *однозадачную* модель, в которой свойство универсальной программируемости замкнуто во внутренних ресурсах компьютера — в оперативном запоминающем устройстве (ОЗУ), арифметико-логическом устройстве (АЛУ), устройстве управления (УУ) и устройстве ввода-вывода (В/В) [5]. Управление вычислениями в рамках этой модели осуществляется устройством управления (УУ) посредством указания *физических* адресов к ячейкам оперативной памяти (ОЗУ), охваченных одномерным, линейно организованным адресным пространством.

В силу изначальной однозадачности в этой модели в принципе отсутствует встроенная универсальная логика защиты памяти от вмешательства одной программы со стороны другой. Отсюда аппаратно незащищенное адресное пространство ОЗУ во многих поколениях компьютеров, реализованных по этой модели.

Многозадачность и другие системные функции управления вычислительными ресурсами, включая защиту памяти от несанкционированного доступа и управление вводом/выводом для всего разнообразия внешней периферии, привносятся программно — на уровне операционных систем (ОС), загружаемых в ОЗУ и функционирующих совместно с многими пользовательскими программами.

В микропроцессорных архитектурах, как известно, используются аппаратные средства поддержки системных функций управления доступом к памяти, которые частично решают и задачи защиты памяти. Но при этом в крайне сложных программных недрах и запутанных лабиринтах ОС остаются множества лазеек для несанкционированного проникновения сторонних программ в области физического адресного пространства опе-

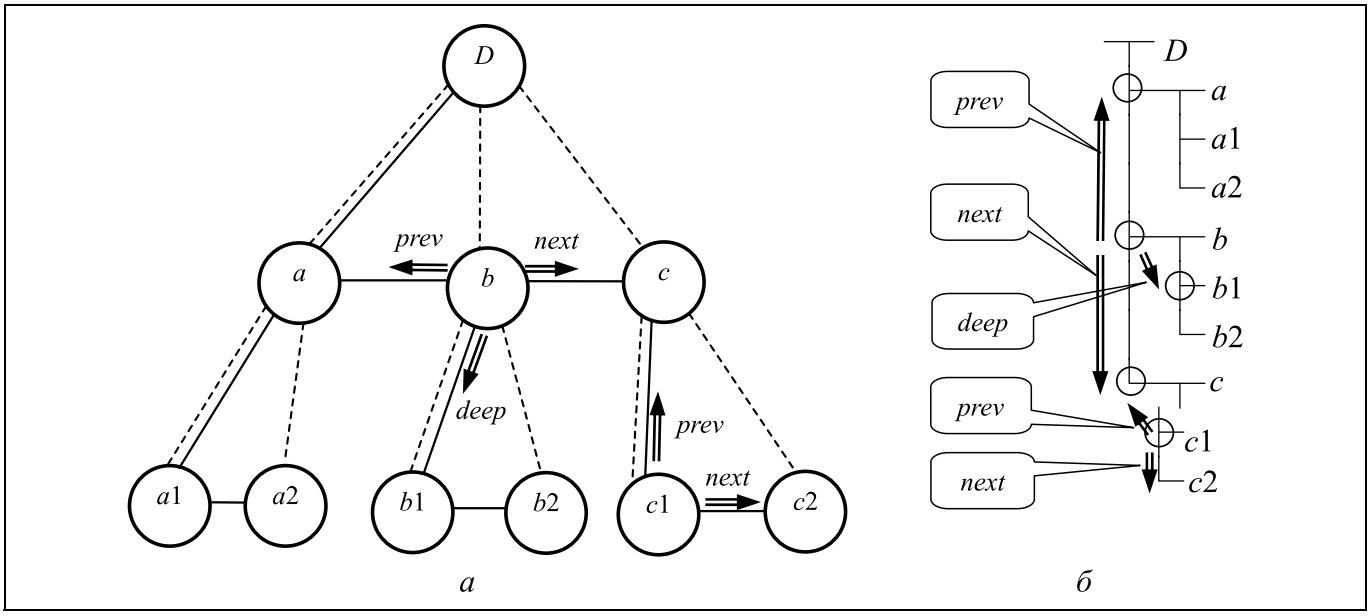

Рис. 1. Древовидная структура (*а*) и компьютерная форма ее представления (*б*)

ративной памяти, что представляет собой трудно контролируемую угрозу безопасности современных компьютеров и систем.

Работа сетей также опирается на многослойные, крайне разнородные системные программы, обеспечивающие реализацию большого числа сетевых протоколов обмена данными.

В чрезмерной сложности разнородных, локализованных и сетевых системных программ кроются причины уязвимости современных компьютерных систем и сетей.

В рамках рассматриваемого общего подхода были выявлены первопричины непрерывного воспроизводства разнородных форм представления компьютерной информации (данных и программ). Они скрыты в постулатах универсального машинного счета классической модели Дж. фон Неймана в виде двух *избыточных степеней свободы управления вычислениями*, открытых для изначально нерегламентированного использования программистами [2—4]. Эта модель позволяет, во-первых, *произвольным образом* выстраивать структуры данных и, во-вторых, *по собственному усмотрению* алгоритмически кодировать их в потоках адресов к физическому адресному пространству ОЗУ.

В указанных степенях свободы скрываются причины *неконтролируемого воспроизводства разнородных форм представления данных, программ, процессов и систем*. Массовое применение компьютеров с микропроцессорными архитектурами, которые имеют еще и разнородные, трудносовместимые между собой аппаратные платформы (что также допускается классической аксиоматикой), привело к закреплению разнородности в масштабах ГКС, что стало одним из главных барьеров фундаментального характера на путях к преодолению ее общесистемного кризиса и дальнейшему сбалансированному развитию [1].

1.2. Компьютерное исчисление древовидных структур

В работах [2—4] посредством математического обобщения классической компьютерной модели проведена ее минимальная коррекция, которая на основе компьютерного исчисления древовидных структур предоставляет математически замкнутую форму регламентации и системы команд, и структур данных, и программ. На уровне обновленной таким образом аксиоматики устраняются избыточные степени свободы, а вместе с ними и первопричины разнородности аппаратных и программных платформ и, как следствие, комбинаторного сопротивления функциональной интеграции ресурсов ГКС.

Математическая регламентация форм представления и способов обработки структурированной информации позволила не только устранить избыточные степени свободы в управлении вычислениями, но и сохранить достоинства классической модели — универсальность и простоту логических правил процедурного управления счетом.

В модели компьютерного исчисления древовидных структур [2—4, 6] универсальный объект исчисления — деревья, которые представляют собой простейшую связную структуру из возможных (число ребер на единицу меньше числа вершин). На рис. 1, *а* показаны деревья общего вида (связи

между вершинами показаны пунктирными линиями), а на рис. 1, *б* —компьютерная форма их представления в виде двоичного дерева (в котором у каждой вершины не более двух соседних снизу вершин).

Двоичные деревья являются универсальным и математически однородным структурным объектом представления программ и данных. Двоичное дерево получается из общего путем устранения всех прямых связей с родительскими вершинами, кроме крайней слева и внесения новых связей между вершинами своего уровня (на рис. 1, *а* связи двоичного дерева показаны сплошной линией).

Содержимое вершин может быть битом, байтом, числом, символом, строкой произвольного размера (двоичной или символьной) или массивом.

1.3. О свойствах процедурного языка ПАРСЕК

На основе нового компьютерного базиса в виде математической модели компьютерного исчисления древовидных структур [3] разработаны процедурный язык ПАРСЕК (ParSeq©) и система программирования на его основе [6].

В языке ПАРСЕК постулируется единое, математически однородное представление и программ, и данных (числового и нечислового — структурного — характера) в виде двоичных деревьев в геометрической форме (рис. 1, *б*).

Древовидные структуры в языке ПАРСЕК изначально не привязаны к какой-либо одной или нескольким предметным областям. Их смысловая интерпретация всецело определяется программистами при решении задач из различных предметных областей.

Посредством древовидных структур в двоичной форме представляются те или иные информационные объекты и структуры, смысловая интерпретация и способы обработки которых определяются программистом в соответствии с решаемой задачей.

Обработка древовидного объекта ведется посредством выделения и обхода вершин двоичных деревьев. Для выделения вершин вводятся *курсорные переменные*, которые принимают значение указателей на вершины, расположенные в памяти. На деревьях можно определять произвольные множества курсорных переменных. Процесс обработки начинается с открытия курсоров и продолжается дальнейшим перемещением их по дереву посредством команд исполняемой программы.

С курсорными переменными связан базисный набор функций для работы с деревьями. Имеется три вида связей для каждой вершины (на рис. 1, *б* показаны утолщенными стрелками):

- *deep* — связь с подчиненной по уровню вершиной;
- *next* — связь со следующей вершиной своего уровня;
- *prev* — связь с предыдущей соседней вершиной своего или старшего уровня.

Компьютерный базис на основе предложенного исчисления — это математически замкнутый и функционально полный на множестве двоичных деревьев набор простейших операций формирования и преобразования двоичных деревьев [2—4, 6]. На этой основе осуществляется минимальная коррекция классической модели фон Неймана путем ее математического обобщения.

Главным объектом математического обобщения классической модели стала оперативная память. Место физического устройства одномерной памяти с произвольным доступом к линейно организованному адресному пространству в обобщенной модели занимает «умная» память хранения множеств произвольных двоичных деревьев с автоматическим их размещением в линейном адресном пространстве ОЗУ. В этой памяти доступ к вершинам деревьев осуществляется посредством курсорных переменных, несущих указатели на вершины, расположенные в памяти.

Управление вычислениями в языке ПАРСЕК осуществляется командами исполняемой программы посредством перемещения курсорных указателей между соседними вершинами и применением к выбранным вершинам вычислительных операций обработки содержимого вершин или функций преобразования (редактирования) деревьев. Таким способом программы на языке ПАРСЕК могут осуществлять произвольные формирования и преобразования древовидных структур.

Особенность языка и системы программирования ПАРСЕК в том, что он в едином математически замкнутом формализме соединяет процедурный (алгоритмический) стиль программирования с возможностями решения задач обработки информации высокой структурно-динамической сложности.

Первым языком высокого уровня стал Fortran (Джон Бэкус, 1954—1957 гг.). Он предназначен для программирования, главным образом, в классе «числовых» задач и реализует процедурный (алгоритмический) стиль программирования (команда за командой), заложенный в модели фон Неймана на машинном уровне. Компиляция программ в машинные коды предполагает статическое распределение памяти, позволяющее сводить к минимуму затраты на перераспределение памяти в ходе исполнения программ.

Вслед за первым процедурным языком высокого уровня был предложен Lisp[1] (Джон Маккарти,

[1] Первоначальную — теоретически целостную версию — называют Pure Lisp (Чистый Лисп).

1958—1963 гг.) — первый функциональный язык программирования [7], основанный на математической теории λ-исчисления Черча. Язык Lisp предназначен для решения задач искусственного интеллекта, особенность которых — ориентация на работу в классах задач обработки «структур» (с информацией высокой структурно-динамической сложности).

По сути, это была первая, практически значимая компьютерная модель, составившая теоретическую альтернативу модели фон Неймана. Процедурному (алгоритмическому) стилю она противопоставила функциональный (непроцедурный) стиль, который требует иных, существенно более сложных механизмов аппаратной реализации. Это стало одной из главных причин того, что модель фон Неймана выиграла соревнование за индустриальное производство массовых компьютеров. Другая причина непопадания в индустрию массового производства компьютеров и программ — относительно малый (на то время) спрос на задачи искусственного интеллекта, а также весьма непростой в использовании формализм, требующий особого склада ума и математической подготовки, что является заведомо избыточным требованием для большинства индустриальных программистов.

Тем не менее, в практическом плане Lisp на многие годы занял свою нишу, предоставляя мощный компьютерный инструментарий для решения логически и структурно сложных задач искусственного интеллекта. В этих задачах доминируют сложные структуры и их преобразование, которые требуют больших затрат времени на динамическое перераспределение памяти, сопровождающееся необходимостью многократного повторения весьма длительных системных процедур сборки мусора [8].

Несмотря на эволюционное сближение возможностей многочисленных поколений языков этих противостоящих классов, разрыв между ними остается непреодоленным. Новая модель процедурных вычислений в базисе исчисления древовидных структур, построенная как математическое обобщение модели фон Неймана, и язык ПАРСЕК на ее основе, позволяют устранить этот разрыв, соединяя достоинства и во многом снимая ограничения принципиально различающихся подходов.

Принципиальная особенность языка ПАРСЕК в том, что в нем осуществлено *математически замкнутое соединение* процедурного стиля программирования с новым компьютерным базисом в виде исчисления древовидных структур. Обновленная классическая модель процедурных вычислений позволила связать в едином формализме достоинства двух ранее противостоящих моделей, до сих пор деливших языки программирования высокого уровня по классам задач обработки «чисел» и «структур».

1.4. Бесшовное программирование компьютеров, связанных сетями

Важнейшим и принципиально новым качеством предлагаемой модели вычислений стала возможность бесшовного распространения свойства универсальной программируемости, замкнутого в классической модели на внутрикомпьютерные ресурсы, на ресурсы сколь угодно большого числа компьютеров, связанных сетями [3, 4, 9].

На рис. 2 представлена виртуальная ПАРСЕК-машина [9], которая иллюстрирует принципы распространения модели исчисления древовидных структур на распределенные вычислительные ресурсы сколь угодно больших сетей.

В рамках такого расширения сформировано единое (сквозное) адресное пространство оперативной памяти компьютеров, связываемых сетями. Оно состоит из двухкомпонентных адресов для идентификации вершин двоичных деревьев <IP:Port, Address>. Первый компонент IP:Port — уникальный сетевой IP-адрес компьютеров, второй — Address — указатель на их расположение в оперативной памяти [9].

При этом данные и программы в виде древовидных структур могут целиком или своими компонентами распределяться по различным ком-

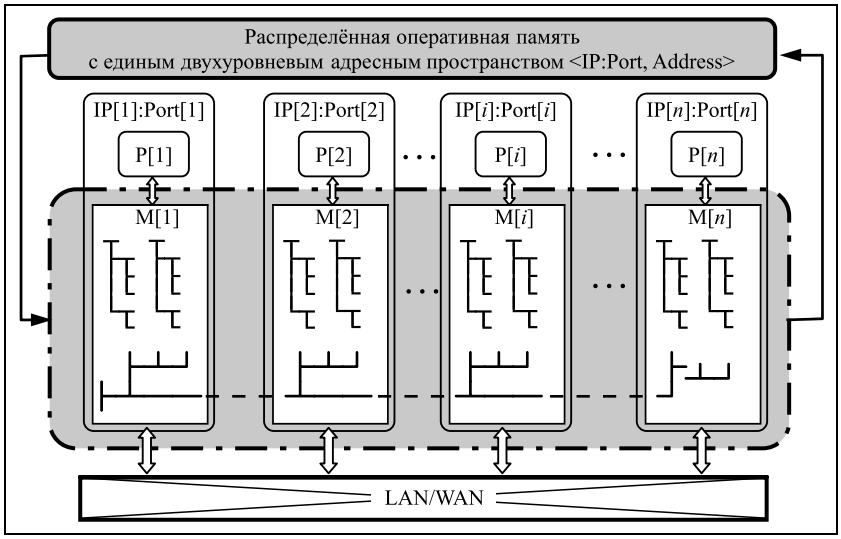

Рис. 2. Виртуальная ПАРСЕК-машина для распределенных вычислений

пьютерам. На рис. 2 показано, как программы и данные в виде древовидных структур могут располагаться в оперативной памяти M[*i*] многих компьютеров с сетевыми адресами IP[*i*]:Port[*i*], охваченной единым адресным пространством. Обработка в каждом *i*-м компьютере осуществляется своими процессорами P[*i*] (параллельно).

Деревья могут располагаться как целиком в локальной памяти компьютеров (на рис. 2 изображены вертикально), так и путем размещения своих связанных компонентов в памяти различных компьютеров (горизонтальное изображение). Важно отметить, что при этом логика работы с деревьями в процессе составления и исполнения программ остается инвариантной относительно способов размещения деревьев в распределенной памяти вовлекаемых компьютеров.

Экспериментальная реализация виртуальной ПАРСЕК-машины для распределенных вычислений осуществлена [9] как функционально полная реализация системы программирования ПАРСЕК с привнесением стандартных команд управления многими процессами и использованием библиотечных функций управления протоколом TCP/IP.

В язык ПАРСЕК добавлены встроенные функции [9, 10], с помощью которых задается распределенная обработка компонентов деревьев посредством одновременного исполнения на разных компьютерах многих асинхронно взаимодействующих процессов. Основными компонентами распределенных вычислений в данной реализации являются как перемещаемые exe-модули, так и подпрограммы с параметрами, которые получили собственную системную реализацию механизма отдаленного запуска процедур RPC (Remote Procedure Calls).

Классические принципы реализации RPC [11] легли в основу технологий распределенных вычислений в сетях. Они предполагают взаимодействие процессов, протекающих в памяти отдаленных компьютеров в отсутствие единого адресного пространства. Такое решение предлагалось в виде технологии «переходного» моста, который посредством устанавливаемого на связанных компьютерах специального промежуточного ПО может перекидываться между компьютерами с разными адресными пространствами.

Главное отличие такой технологии от типового локального вызова процедур, реализуемого во всех процедурных языках программирования, состоит в том, что передача параметров между отдаленно взаимодействующими процессами при запуске процедур не допускает передачу данных по адресным ссылкам и требует обязательного копирования пересылаемых значений.

В известных реализациях механизмов RPC отсутствуют возможности работы с глобальными переменными и передачи адресных указателей в качестве параметров процедур. Неустраненные логические различия в вызове локальных и отдаленных процедур создают технологические проблемы достижения «прозрачности» программирования распределенных вычислений.

В отсутствие единого адресного пространства, охватывающего оперативную память компьютеров в сетях, и единой универсальной модели распределенных вычислений механизмы RPC вынужденно реализовывались на уровне частных способов стыковки разнородных адресных пространств посредством сложных промежуточных программных слоев, которые повышают степень разнородности ГКС.

Такие ограничения приводят к тому, что отдаленно вызываемые процедуры обладают меньшими функциональными возможностями, чем локальные.

Последующее развитие технологии отдаленного взаимодействия программных процессов шло в направлениях поддержки индустриально значимых языков объектно-ориентированного программирования. Это такие технологии, как CORBA [12], DCOM [13], а также средства создания на их основе межплатформенных технологий интеграции сложных распределенных программных комплексов, например, такие как SOA [14].

Реализация механизма RPC в системе ПАРСЕК отличается от известных типовых подходов. Во-первых, тем, что он реализован в едином адресном пространстве, охватывающем оперативную память связанных сетями компьютеров, вовлекаемых в распределенные вычисления. При этом передача параметров при вызове как локальных, так и отдаленных процедур происходит по одинаковым правилам, что делает работу в сети полностью прозрачной. Во-вторых, эта реализация встроена в систему ПАРСЕК как внутренний механизм и не предназначается, в отличие от существующих решений, к самостоятельному использованию в условиях легализованной в ГКС разнородности аппаратных и программных платформ. В системе ПАРСЕК это позволило существенно упростить программную реализацию RPC и открыло возможности распространения свойства бесшовной программируемости в нотациях языка исчисления древовидных структур на сколь угодно большие сети.

Встраиваемые в ПАРСЕК механизмы распространения свойства универсальной программируемости на сетевые ресурсы логически и технологически прозрачны для программистов и не

требуют сложного системно-сетевого конфигурирования и сопровождения.

Язык и система ПАРСЕК рассматриваются как прототип бесшовного программирования в математически однородном алгоритмическом пространстве распределенных вычислений и СЦУ.

В этом пространстве отпадает необходимость в сторонних технологиях функциональной интеграции изначально разнородных сетевых ресурсов. Предполагается, что внутренние механизмы управления бесшовно программируемыми распределенными вычислениями, включая собственное воплощение RPC, задействованные в системе ПАРСЕК, будут реализованы аппаратно в сетевых однокристальных компьютерах с немикропроцессорной архитектурой (см. далее), система команд которых строится на основе обновленной классической модели в базисе исчисления древовидных структур.

Система программирования ПАРСЕК с функциями распределенных вычислений в едином алгоритмическом пространстве исчисления древовидных структур испытана на задаче поиска оптимальных конфигураций коммутирующих сетей на основе комбинаторных методов построения квазиполных графов [15]. Алгоритм решения этой задачи допускает разбиение на многие слабо связанные фрагменты, каждый из которых требует большого объема вычислений. Это позволяет исполнять их одновременно на многих компьютерах, связанных через сети.

Применительно к данной задаче с помощью системы ПАРСЕК разработана стендовая система распределенных вычислений в математически однородном алгоритмическом пространстве [16], в которой в распределенные вычисления вовлекались десятки компьютеров как из локальных сетей нескольких организаций, так и домашних. Принципиальных ограничений на число вовлекаемых компьютеров нет. Взаимодействие компьютеров через Интернет (в том числе обмены данными) осуществляется в логике сетевой вычислительной архитектуры «Peer-to-Peer». При этом возможен запуск с любого числа компьютеров и одновременное исполнение многих задач с разными значениями параметров их размерностей и конфигураций разбиения на фрагменты.

Для присоединения компьютеров к распределенной системе вычислений требуется регистрация на сайте системы их IP-адресов и последующая установка небольшой программы, обеспечивающей взаимодействие компьютеров в ходе распределенных вычислений.

Эксперименты показали, что на задаче, разбиваемой на большое число фрагментов со значительными объемами вычислений, требующими заведомо большего времени, чем время обменов по сети промежуточными данными, обеспечивается сокращение общего времени счета, пропорциональное числу вовлеченных компьютеров.

Ключевой результат: новый компьютерный базис исчисления древовидных структур позволяет бесшовно распространить свойство универсальной программируемости с внутрикомпьютерных ресурсов на любые совокупности связанных сетями компьютеров, что становится основой для формирования в сколь угодно больших сетях универсального, математически однородного и бесшовно программируемого алгоритмического пространства распределенных вычислений и СЦУ.

Наследуя универсальность и простоту процедурного стиля программирования классической компьютерной аксиоматики, новая модель распределенных вычислений позволяет интегрировать в новое алгоритмическое пространство функциональные возможности и наработки имеющихся в ГКС компьютерных и программных платформ, обеспечивая тем самым эволюционный переход в новое алгоритмическое пространство, охватывающее совокупные ресурсы ГКС.

В математически однородном алгоритмическом пространстве открываются качественно новые возможности для решения непрерывного спектра задач СЦУ высокой структурной сложности с привлечением совокупного системообразующего потенциала ГКС.

Это необходимо для *массовой интеллектуализации* всего разнообразия процессов управления устойчивым развитием социосистем в условиях глобальной сильносвязности [1]. Это актуально для построения разных видов систем СЦУ военного назначения [17], а также решения важнейших для разнообразных видов бизнеса классов задач массовой переработки глобально распределенной информации, рассмотренных в работе [1] — Internet of Things [18], Internet of Everything [19] и др.

2. АРХИТЕКТУРНО-АППАРАТНАЯ ПОДДЕРЖКА БЕСШОВНО ПРОГРАММИРУЕМОГО И КИБЕРБЕЗОПАСНОГО АЛГОРИТМИЧЕСКОГО ПРОСТРАНСТВА

Бесшовно программируемое алгоритмическое пространство дает стратегическое направление для качественного совершенствования ГКС, нацеленное на устранение разнородности и других фундаментальных барьеров, препятствующих снижению ее общесистемной сложности.

Необходимое условие реализуемости единого алгоритмического пространства — наличие эффективных архитектурно-аппаратных решений, обеспечивающих компьютерно-сетевую реализа-

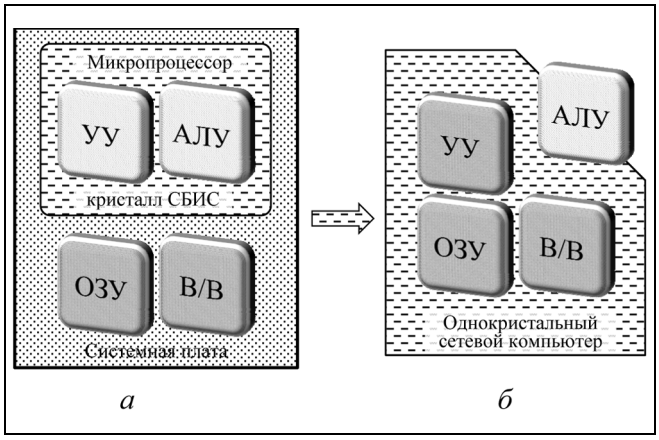

Рис. 3. От микропроцессорной архитектуры (*а*) к «немикропроцессорной» (*б*)

цию системных функций, поддерживающих исполнение операций математически замкнутого базиса преобразования деревьев.

Ключевые системные функции, образующие системный базис управления машинными ресурсами, предлагается реализовать на аппаратном уровне посредством универсальных сетевых компьютеров с новой — немикропроцессорной — архитектурой [3, 20, 21] (рис. 3).

Из четырех блоков АЛУ, ОЗУ, УУ и В/В фоннеймановских универсальных компьютеров [5] микропроцессоры в одном кристалле СБИС реализуют (рис. 3, *а*) два — АЛУ и УУ. Память (ОЗУ) и устройства ввода-вывода (В/В) вынесены как отдельные блоки, управление которыми осуществляет УУ из микропроцессора. Включение УУ в состав микропроцессора позволяет характеризовать микропроцессорные архитектуры компьютеров как («умная» арифметика) & («глупая» память).

Известно, что до сих пор системный интеллект компьютеров реализуется программно посредством ОС, загружаемых в оперативную память. Несбалансированность микропроцессорной архитектуры выражается в том, что управление вычислениями осуществляется из УУ микропроцессора, не обладающего собственной системной памятью для хранения текущего состояния вычислительных процессов. Отсюда высокие накладные расходы — чрезмерная сложность ОС, большие расходы памяти для ОС, большие временные затраты на последовательную реализацию многошаговых системных команд, слабая защищенность системных областей оперативной памяти от несанкционированного вмешательства других программ и др.

Компьютерный базис исчисления древовидных структур позволяет относительно просто (благодаря математической полноте и замкнутости) переходить к системно сбалансированной и более эффективной — немикропроцессорной — архитектуре компьютеров (рис. 3, *б*). Система команд компьютеров с немикропроцессорной архитектурой строится на основе нового компьютерного базиса исчисления древовидных структур [3].

Такие архитектуры открывают новый класс массовых компьютерных устройств, которые предназначаются для использования как универсальные однокристальные сетевые узлы, обеспечивающие формирование единого, математически однородного, бесшовно программируемого и кибербезопасного алгоритмического пространства, с качественно новыми системообразующими свойствами, о которых говорилось выше, а также в работе [1].

Особенность и основа архитектуры однокристального сетевого компьютера с немикропроцессорной архитектурой (рис. 3, *б*) — «умная» оперативная память большого объема со встроенным «системным интеллектом». Он реализуется аппаратно в СБИС «умной» памяти посредством УУ с эффективным алгоритмом управления, воплощающим набор операций нового компьютерного базиса. Арифметические возможности наращиваются путем внешнего подключения ведомых АЛУ (рис. 3, *б*). В качестве таких «сопроцессоров» могут использоваться разные вычислители — от специальных ускорителей и универсальных микропроцессоров до суперкомпьютеров. Это позволяет характеризовать немикропроцессорную архитектуру компьютеров как («умная» память) & («глупая» арифметика).

В компьютерном базисе исчисления древовидных структур становится возможным эффективный перенос ключевых системных функций, которые до сего времени программно реализуются в ядре ОС, а также в сетевом оборудовании для поддержки сетевых протоколов, на аппаратный уровень. «Умная» память аппаратно (на транзисторном уровне СБИС) реализует функции ядра ОС и обеспечивает:

- автоматическое управление динамическим распределением памяти при размещении в ней и удалении элементов преобразуемых древовидных структур (включая сборку мусора);
- многозадачный режим, виртуализацию памяти;
- управление вводом-выводом;
- выход в сети и сетевые взаимодействия по осуществлению бесшовно программируемых распределенных вычислений;
- полную, встроенную в память, аппаратную защиту от несанкционированного доступа к физическому адресному пространству ОЗУ со стороны исполняемых программ.

Благодаря аппаратной реализации системных функций достигается предельно высокие уровни эффективности по управлению и защите внутрикомпьютерных и сетевых ресурсов, вовлекаемых в распределенные вычисления.

Обеспечение кибербезопасности — это не только функциональная защита от зловредных программ, но и, прежде всего, борьба с причинами чрезмерной системно-технической сложности, которая приводит к утрате контроля над внутренними степенями свободы больших систем и росту уязвимостей компьютерной среды [21].

Важнейшая особенность нового пространства, открывающая возможности кардинального снижения системотехнической сложности ГКС, — разнесение системных и прикладных функций на разные уровни: системные выносятся на аппаратный, прикладные остаются на программном.

Размещение и обработка компонентов древовидных структур в распределенной оперативной памяти компьютеров, связанных сетями, в ходе исполнения прикладных программ осуществляется автоматически посредством аппаратно реализованных системных функций управления машинными/сетевыми ресурсами, которые полностью скрыты от программиста под математически замкнутыми операциями преобразования древовидных структур.

Все операции преобразования древовидных структур в математически замкнутом базисе исчисления древовидных структур трактуются как инструмент решения прикладных задач. Воплощая математически замкнутую логику произвольных алгоритмических преобразований сколь угодно больших древовидных структур, они составляют машинонезависимый уровень программирования прикладных задач, на котором логика обработки древовидных структур инвариантна относительно сетевых конфигураций используемых вычислительных ресурсов.

Как следствие, это сводит к минимуму необходимость «ручного» управления взаимодействием прикладных задач с системной программной средой, которое типично для индустриально значимых языков программирования.

Новое алгоритмическое пространство — это не только путь к повышению эффективности и надежности исполнения системных функций управления вычислениями в ГКС на порядки. Это — избавление от необходимости создания новых многослойных, крайне разнородных, непомерно раздувшихся и, потому, трудно контролируемых программных реализаций системных функций (ОС, промежуточное ПО, сетевые протоколы). Это — возможность взаимодействия и совместного функционирования в ГКС нового алгоритмического пространства и всех наработок в существующих программных платформах.

На этом пути открываются возможности для прекращения роста разнородности ГКС, для обеспечения в дальнейшем кардинального снижения степени системной разнородности совокупных ресурсов ГКС, а значит и чрезмерной, все хуже контролируемой системотехнической сложности ГКС, что позволит вывести ее на качественно новые уровни эффективности и кибербезопасности.

Важно отметить, что аппаратная реализация системных функций в «умной» памяти, построенная на основе математически замкнутых моделей, не имеет «неучтенных» степеней свободы и «скрытых функций» управления, что крайне важно для обеспечения абсолютной защиты от несанкционированного вмешательства на системные уровни со стороны прикладных программ. Защита обеспечивается тем, что со стороны прикладных программ в новой архитектуре отсутствуют каналы нелегального доступа (в обход математически замкнутых операций исчисления древовидных структур) к внутренним механизмам системных уровней, аппаратно реализованным в «умной» памяти.

В рамках новой (немикропроцессорной) архитектуры и единого алгоритмического пространства открываются возможности общего решения проблем обеспечения кибербезопасности ГКС в разных сферах:

- безопасность прикладных сервисов — благодаря решению прикладных задач в математически замкнутом базисе исчисления древовидных структур;
- системотехническая безопасность — благодаря:
  - — исполнению на аппаратном уровне математически верифицированных системных функций управления вычислительными ресурсами и процессами;
  - — полной защите физического адресного пространства оперативной памяти от несанкционированного доступа со стороны собственных и сторонних вычислительных процессов;
  - — исключению необходимости создания новых промежуточных программных слоев (ОС, поддержка протоколов разных уровней и др.);
- информационная безопасность — благодаря новым возможностям перехода к высокоорганизованным формам накопления и систематизации глобально распределенной информации посредством глубокого структурирования данных и программ в математически замкнутом базисе исчисления древовидных структур;
- социальная безопасность [1] — благодаря высокой степени структурирования глобально распределенной информации, позволяющей уст-

ранять «токсичные» и деструктивные воздействия сверхпотоков плохо организованной информации на человека и социумы.

3. ПУТИ РЕШЕНИЯ ПРОБЛЕМ ОРГАНИЗАЦИИ НАДЕЖНЫХ ВЫЧИСЛЕНИЙ

Огромное количество компьютеров ГКС и наличие большой избыточности связей между ними дает необходимый резерв для программных методов обеспечения надежного выполнения распределенных вычислений и процессов СЦУ в вычислительных средах с ненадежными компонентами. Ненадежность компонентов обусловлена неустранимой недетерминированностью вхождения вовлекаемых вычислительных узлов и связей сетевых ресурсов. Причины недетерминированности — негарантированное включение отдаленных устройств, поломки, сбои, деструктивные воздействия и др.

В общем случае доступ пользователей своих компьютерных устройств к аппаратным и системотехническим уровням повышения надежности чужих компьютеров и сетевого оборудования ГКС отсутствует. При этом каждому пользователю необходимо предоставить высокую надежность распределенного исполнения его программ при том состоянии компьютерной среды, которое де-факто имеется на каждый текущий момент.

В отсутствие на системотехнических уровнях внешнего доступа к отдаленным ресурсам сетей наиболее доступным путем к обеспечению надежности исполнения программ каждого пользователя-программиста становится применение программных методов подготовки прикладных программ распределенных вычислений к исполнению в компьютерной среде с отработкой задаваемых уровней параллелизма.

Бесшовная программируемость единого алгоритмического пространства распределенных вычислений открывает возможности обеспечения требуемой надежности исполнения программ путем автоматического внесения информационной, коммуникационной и вычислительной избыточности в программы пользователей на этапах их трансляции и последующей компоновки исполняемых машинных кодов. При этом учитывается наличие в программах параллельно исполняемых фрагментов, которые в ходе исполнения на многих компьютерах напрямую обмениваются промежуточными данными.

Информационная избыточность реализуется разбиением массивов данных и файлов на порции, с последующим их копированием и распределенным хранением в памяти различных компьютеров. Коммуникационная избыточность — одновременной передачей идентичных копий порций данных от мест их хранения к местам их обработки по разным маршрутам. Вычислительная избыточность — посредством одновременного исполнения на разных компьютерах многих копий идентичных для каждого фрагмента задачи процессов с обеспечением обменов данных между ними через сети по многим маршрутам.

Вычислительная избыточность вносится на этапах трансляции прикладных программ автоматически посредством специальных вставок в коды машинных программ. Сохраняя правильное функционирование и требуемые уровни параллелизма, они осуществляют надежное исполнение программы при наличии больших резервов ресурсов путем привлечения (через сети) в ходе вычислений любого требуемого для сохранения заданных уровней параллелизма и надежности количества работоспособных компьютерных устройств.

Повышение надежности распределенного исполнения программ с привнесенной избыточностью обеспечивается путем одновременного (параллельного) выполнения многих копий машинных кодов программ с идентичными начальными состояниями и входными данными на многих компьютерных устройствах. В системе ПАРСЕК фрагменты программ, с указанным параллелизмом их исполнения, реализуются на доступных компьютерах как запуск процедур с отдаленным доступом — RPC (см. выше).

Надежность достигается посредством исполнения в идентичных копиях программ процедуры мажорирования, которая встраивается в коды исполняемых программ на этапе трансляции-компоновки. Она привязывается к идентификаторам контрольных точек, расставленных в программе. В ходе исполнения программы во всех одноименных контрольных точках всех ее копий процессов, исполняемых на различных компьютерах, осуществляется их приостановка с последующими обменами промежуточными данными между всеми копиями в целях взаимного сравнения и мажорирования текущего состояния процессов в контрольных точках.

Совпадение большинства промежуточных результатов указывает на то, что компьютеры, в которых они получены, находятся в работоспособном состоянии. Отсутствие ожидаемых сведений или несовпадение с результатом большинства говорит о неработоспособности соответствующих компьютеров и необходимости их замены, с передачей копии текущего состояния правильного процесса на другой компьютер сети из пула дееспособных.

В математически однородном пространстве исчисления древовидных структур пулы дееспособных компьютеров в масштабах ГКС априори не ограничены. В отсутствие разнородности форм представления данных и программ проблемы перекодировок для сравнения промежуточных результатов и передачи состояния копий правильных процессов на дееспособные компьютеры не возникают.

В предполагаемом подходе представлена программная реализация модели резервирования с заменой (хорошо известной в теории надежности) в ее адаптации к особенностям ГКС.

Исследования математической модели рассмотренного способа обеспечения надежности показали [22] экспоненциальный характер роста надежности при линейном увеличении количества идентичных процессов. Так, внесение избыточности сравнительно небольшой кратности ~5 может повысить надежность на многие порядки. Полученные оценки дают количественные зависимости между требуемой надежностью исполнения программ пользователей и кратности внесения на этапе их трансляции-компоновки необходимой избыточности.

Изложенный способ обеспечения надежности частично реализован в рамках технологии ПАРСЕК в представленной выше стендовой системе распределенных вычислений в математически однородном алгоритмическом пространстве [16].

ЗАКЛЮЧЕНИЕ

Во второй части работы продолжено исследование общесистемных свойств глобальной компьютерной среды (ГКС) — принципиально нового, сверхбольшого и глобально сильносвязного кибернетического объекта, введенного в рассмотрение в статье [2] первой части работы [1], который оказывает беспрецедентное по масштабам, глубине и темпам влияние на мировую социосистему. Стихийный и сверхбыстрый рост крайне разнородной ГКС привел к массовым системотехническим противоречиям и общесистемному кризису в ее развитии, который проявляется в экспоненциальном росте слабо формализованной информации, непригодной для осуществления качественного управления.

Глубокий внутренний кризис ГКС в нарастающих масштабах оказывает негативное влияние на устойчивость развития социосистем, которое сопровождается чередой глобальных кризисов, не поддающихся известным методам и способам политического и экономического регулирования.

Для восстановления управляемости социосистем в условиях глобальной сильносвязности необходим адекватный масштабам возникающих проблем глобально распределенный инструмент управления устойчивым развитием. Таким инструментом может стать только ГКС, обладающая практически неограниченным потенциалом наращивания вычислительных ресурсов и функциональных возможностей, необходимых для полномасштабной переработки экспоненциально растущих потоков и объемов информации в целях управления устойчивым развитием социосистем.

Необходимое условие для этого — принципиальное обновления системообразующего потенциала ГКС.

Представленный подход к формированию в ГКС математически однородного алгоритмического пространства показывает пути к общему решению проблем программируемости, безопасности и надежности распределенных вычислений и сетецентрического управления (СЦУ).

Научная новизна подхода начинается с пересмотра базовых принципов организации машинных вычислений, заложенных в классической модели Дж. фон Неймана. Математическое обобщение классической компьютерной аксиоматики на основе исчисления древовидных структур открывает возможности:

- устранения первопричин воспроизводства разнородности аппаратных и программных платформ;
- распространения свойства универсальной программируемости с внутрикомпьютерных ресурсов на любые совокупности компьютеров, связанных сетями;
- построения нового класса сетевых универсальных компьютеров, имеющих немикропроцессорную архитектуру;
- формирования в сколь угодно больших сетях на основе компьютеров с немикропроцессорной архитектурой бесшовно программируемого и кибербезопасного алгоритмического пространства надежных распределенных вычислений и процессов СЦУ.

Практическое воплощение изложенных принципов и методов формирования математически однородного алгоритмического пространства открывает пути к созданию принципиально новых технологий решения в ГКС с минимальными издержками всего разнообразия задач распределенных вычислений и СЦУ устойчивым функционированием и развитием больших распределенных систем в условиях глобальной информационной сильносвязности.

Это впрямую относится к технологиям общего и высокорентабельного решения комплексных

проблем СЦУ как в военных сферах, так в разнообразных сферах управления функционированием и развитием социосистем. Ключевую роль такие технологии сыграют в опережающем решении обширной проблематики, обозначенной в концепциях Internet of Things [18] и Internet of Everything [19], которые открывают на компьютерном рынке новую нишу емкостью $14.4 трлн. до 2022 г.

ЛИТЕРАТУРА

1. *Затуливетер Ю.С., Фищенко Е.А.* Проблемы программируемости, безопасности и надежности распределенных вычислений и сетецентрического управления. Ч. 1. Анализ проблематики // Проблемы управления. — 2016. — № 3. — С. 49—57.
2. *Затуливетер Ю.С.* Проблемы глобализации парадигмы управления в математически однородном поле компьютерной информации. Ч. 1; Ч. 2 // Проблемы управления. — 2005. — № 1. — С. 2—10; № 2. — С. 12—23.
3. *Затуливетер Ю.С.* Компьютерный базис сетецентрического управления // Тр. конф. «Технические и программные средства систем управления, контроля и измерения» УКИ'10 / ИПУ РАН. — М., 2010. — С. 17—37. — URL: http://www.ipu.ru/sites/default/files/publications/38190/20052-38190.pdf (дата обращения: 22.06.2016).
4. *Затуливетер Ю.С.* К новой компьютерной аксиоматике // Тр. третьей междунар. конф. «Идентификация систем и задачи управления», SICPRO`04 / ИПУ РАН. 28—30 янв. 2004 г. — М., 2004. — С. 2187—2193.
5. *Беркс А., Голдстейн Г., Нейман Дж.* Предварительное рассмотрение логической конструкции электронного вычислительного устройства // Кибернетический сборник. — 1964. — Вып. 9. — С. 7—67.
6. *Затуливетер Ю.С., Халатян Т.Г.* ПАРСЕК — язык компьютерного исчисления древовидных структур с открытой интерпретацией. Стендовый вариант системы программирования. — М.: ИПУ РАН, 1997. — 71 с.
7. *McCarthy, John.* Recursive Functions of Symbolic Expressions and Their Computation by Machine, Part I // Communications of the ACM. — 1960. — Vol. 3, N 4. — P. 184—195. — URL: http://www-formal.stanford.edu/jmc/recursive.pdf (дата обращения: 22.06.2016).
8. *Кнут Д.Э.* Искусство программирования. Т. 1. Основные алгоритмы. — М.: Изд. дом «Вильямс», 2000. — 720 с.
9. *Затуливетер Ю.С., Топорищев А.В.* Язык ПАРСЕК: программирование глобально распределенных вычислений в модели исчисления древовидных структур // Проблемы управления. — 2005. — № 4. — С. 12—20.
10. *Затуливетер Ю.С., Фищенко Е.А.* Организация распределенных вычислений в системе программирования ПАРСЕК на примере сжатия цифрового видео // Проблемы управления. — 2003. — № 4. — С. 6—10.
11. *Birrell A.D., Nelson B.J.* Implementing remote procedure calls // ACM Trans. Comp. Systems. — 1984. — Vol. 2, N 1. — P. 39—59. — URL: http://birrell.org/andrew/papers/ImplementingRPC.pdf (дата обращения: 22.06.2016).
12. *Орфали Р., Харки Д., Эдвард Д.* Основы CORBA. — М.: Малип, 1999. — 316 с.
13. *Модель* COM/DCOM. — URL: http://www.interface.ru/fset.asp?Url=/borland/com_dcom.htm (дата обращения 12.07.2016).
14. *Service* Oriented Architecture (SOA). — URL: http://www.interface.ru/home.asp?artId=3352 (дата обращения 22.06.2016).
15. *Каравай М.Ф., Пархоменко П.П., Подлазов В.С.* Комбинаторные методы построения двудольных однородных минимальных квазиполных графов (симметричных блок-схем) // Автоматика и телемеханика. — 2009. — № 2. — С. 153—170.
16. *Артамонов С.Е., Затуливетер Ю.С., Козлов В.А.* и др. Экспериментальный стенд распределенных вычислений в математически однородном алгоритмическом пространстве // Тр. конф. «Технические и программные средства систем управления, контроля и измерения» УКИ-2012 / ИПУ РАН. — М., 2012. — С. 001956—001961. — URL: http://www.ipu.ru/sites/default/files/publications/16726/2885-16726.pdf (дата обращения: 22.06.2016).
17. *Затуливетер Ю.С., Семенов С.С.* Ориентир — достаточная оборона // Национальная оборона. — 2012. — № 11 (80). — С. 30—40. — URL: http://www.oborona.ru/includes/periodics/conceptions/2012/1112/14319513/detail.shtml (дата обращения: 22.06.2016).
18. *Dave Evans.* The Internet of Things How the Next Evolution of the Internet is Changing Everything. 2011. — URL: http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf (дата обращения: 22.06.2016).
19. *Bradley J., Barbier J., Handler D.* Embracing the Internet of Everything to Capture your Share of $14,4 trillion. CISCO, White Paper. 2013. — URL: http://www.cisco.com/web/PH/ciscoinnovate/pdfs/IoE_Economy.pdf (дата обращения: 22.06.2016).
20. *Затуливетер Ю.С., Фищенко Е.А., Семенов С.С.* Принципы формирования универсального алгоритмического пространства распределенных и параллельных вычислений на основе немикропроцессорных компьютерно-сетевых архитектур // Вестник компьютерных и информационных технологий. — 2013. № 6 (108). — С. 3—10. — URL: http://www.ipu.ru/sites/default/files/publications/25296/7293-25296.pdf (дата обращения: 22.06.2016).
21. *Затуливетер Ю.С., Фищенко Е.А.* Принципы формирования универсального бесшовно программируемого и кибербезопасного алгоритмического пространства // Программные системы: теория и приложения. — 2014. — № 1. — С. 153—173. — URL: http://psta.psiras.ru/read/psta2014_1_153-173.pdf (дата обращения: 22.06.2016).
22. *Затуливетер Ю.С., Топорищев А.В., Фищенко Е.А., Ходаковский И.А.* Принципы реализации моделей повышения надежности распределенных вычислений в системе программирования ПАРСЕК // Датчики и системы. — 2009. — № 12. — С. 11—16. — URL: http://www.ipu.ru/sites/default/files/publications/7192/1206-7192.pdf (дата обращения: 22.06.2016).

Статья представлена к публикации членом редсовета чл.-корр. РАН П.П. Пархоменко.

Затуливетер Юрий Семенович — канд. техн. наук, вед. науч. сотрудник, ✉ zvt@ipu.rssi.ru,

Фищенко Елена Алексеевна — канд. техн. наук, вед. науч. сотрудник, ✉ elena.fish@mail.ru,

Институт проблем управления им. В.А. Трапезникова РАН, г. Москва.