УДК 004.457+004.031.2+004.382.2
DOI: 10.15827/0236-235X.128.581-594
Дата подачи статьи: 18.07.19
2019. Т. 32. № 4. С. 581–594

Методы и средства моделирования системы управления суперкомпьютерными заданиями

А.В. Баранов [1], *к.т.н., доцент, ведущий научный сотрудник, antbar@mail.ru, abaranov@jscc.ru*
Д.С. Ляховец [2], *научный сотрудник, anetto@inbox.ru*

[1] *Межведомственный суперкомпьютерный центр РАН – филиал Федерального научного центра Научно-исследовательский институт системных исследований РАН, г. Москва, 119334, Россия*
[2] *Научно-исследовательский институт «Квант», г. Москва, 125438, Россия*

В статье рассматриваются методы и средства моделирования систем управления суперкомпьютерными заданиями (СУЗ), таких как SLURM, PBS, Moab и отечественная система управления прохождением параллельных заданий (СУППЗ).

Среди методов моделирования СУЗ выделены натурный эксперимент, моделирование СУЗ с виртуальным вычислителем, имитационное моделирование. Рассмотрены методы и способы построения модельного потока заданий. На примере СУППЗ показана невозможность точного воспроизведения натурного эксперимента.

Поставлен вопрос об адекватности модели СУЗ, введены понятия адекватности в широком и узком смыслах. Показано, что адекватная в узком смысле модель СУЗ обеспечивает соответствие только интервальных показателей и не может быть использована в качестве прогнозной модели. Для определения адекватности в широком смысле рассмотрена численная оценка близости двух потоков событий СУЗ – реального и полученного в результате моделирования. В качестве меры близости двух потоков предложено нормализованное евклидово расстояние между двумя векторами, соответствующими сравниваемым потокам. Размерность векторов равна числу обработанных заданий, а компоненты векторов представляют собой времена пребывания заданий в системе.

Для меры адекватности предложена методика ее определения, основанная на сравнении статистики работы реальной системы и модели СУЗ. На примере СУППЗ определено эталонное значение меры адекватности как нормированное евклидово расстояние между векторами времен пребывания заданий в системе, полученными от реальной СУППЗ и модели СУППЗ с виртуальным вычислителем.

Ключевые слова: *высокопроизводительные вычисления, системы управления заданиями, планирование суперкомпьютерных заданий, имитационное моделирование, адекватность модели.*

Типовым режимом функционирования современных суперкомпьютеров является режим коллективного пользования. Множество пользователей разделяют между собой решающее поле суперкомпьютера, которое образуют объединенные высокопроизводительной сетью *вычислительные модули* (ВМ). Как правило, у пользователей нет непосредственного доступа к решающему полю, для производства расчетов на суперкомпьютере они должны сформировать т.н. вычислительное задание, состоящее из расчетной программы, исходных данных и требований к объему (число процессорных ядер или ВМ, размер оперативной памяти) вычислительных ресурсов и времени выполнения задания. Формализованное описание задания часто называют паспортом задания.

Управляют заданиями в суперкомпьютерах специальные программные системы [1], такие как SLURM [2], PBS [3] или российская *система управления прохождением параллельных заданий* (СУППЗ) [4]. Главным компонентом, ядром любой *системы управления заданиями* (СУЗ) является планировщик, отвечающий за ведение очереди заданий. Планировщик на основе информации из паспортов заданий, главным образом, информации о требуемых виде, объеме ресурсов и времени выполнения, формирует расписание запусков заданий. В соответствии с этим расписанием планировщик СУЗ выдает достаточно точный прогноз времени запуска каждого находящегося в очереди задания. Изменения в этот прогноз обычно вносятся при перестроении расписания по причине поступления в систему нового задания, удаления задания из очереди либо преждевременного завершения выполняющегося задания.

Для оценки качества планирования и составления расписания запусков заданий используется ряд показателей, таких как загрузка

решающего поля, среднее время ожидания задания в очереди и другие [5]. На эти показатели существенное влияние оказывают как параметры конфигурации планировщика, так и характеристики входного потока заданий. При этом далеко не всегда это влияние очевидно и может быть рассчитано или предсказано, поскольку современные СУЗ – достаточно сложные системы, обладающие множеством настраиваемых параметров. Это обусловливает актуальность задачи моделирования работы СУЗ с целью исследования влияния параметров входного потока и конфигурации СУЗ на показатели качества планирования заданий.

Устойчивой тенденцией развития высокопроизводительных вычислений является объединение территориально распределенных суперкомпьютерных ресурсов [6]. Главная цель подобного объединения в последние годы – создание инфраструктурных и прикладных цифровых платформ для обеспечения потребностей организаций науки, образования и промышленности в высокопроизводительных вычислениях. Системы управления заданиями суперкомпьютеров входят в состав таких цифровых платформ в качестве отдельных компонентов. В состав одной цифровой платформы могут входить несколько СУЗ, при этом появляется множество входных потоков заданий, каждое из которых в общем случае может быть распределено в любую СУЗ из состава платформы. Как следствие, многократно возрастают сложность управления заданиями и сложность прогнозирования времени и места запуска задания. Под местом запуска здесь понимается суперкомпьютер из состава распределенной цифровой платформы, на котором будет выполнено задание. Для эффективного планирования заданий в распределенной системе необходимо точно прогнозировать время освобождения требуемых вычислительных ресурсов в каждой из СУЗ [7], что может быть достигнуто путем моделирования системы управления с целью формирования прогноза времени и места запуска каждого задания.

Любая модель, в том числе прогнозная, представляет собой некоторое абстрактное отображение реально функционирующей системы. Неизбежное при создании модели абстрагирование приводит к тому, что модель по определению не может во всей полноте и точности воспроизвести моделируемую систему. Иными словами, поведение модели почти всегда будет несколько отличаться от поведения реальной системы, что порождает две взаимосвязанные задачи. Во-первых, необходимо знать, каким образом можно измерить точность воспроизведения моделью поведения моделируемой системы, то есть определить меру точности (адекватности) модели. Подобная мера позволит сравнивать различные модели между собой по степени адекватности. Во-вторых, необходимо установить предельно допустимое значение (границу) меры точности, выход за которое будет означать, что модель недостаточно адекватна и не может быть использована для анализа поведения реальной системы. Исследование методов и средств моделирования СУЗ, выбор меры адекватности, а также определение границ адекватности для моделей систем управления суперкомпьютерными заданиями являются задачами настоящей работы.

Задача моделирования СУЗ суперкомпьютера

Любая СУЗ суперкомпьютера выполняет следующие функции:

– отвечает за прием различных заданий от разных пользователей;

– обеспечивает ведение очереди заданий (планирование заданий);

– выделяет очередному заданию в соответствии с его требованиями подмножество ВМ решающего поля и осуществляет конфигурацию этого подмножества;

– производит запуск задания на выделенных ему ВМ решающего поля;

– контролирует выполняющееся задание, завершая его при необходимости по истечении заказанного времени обработки;

– освобождает выделенные ресурсы после завершения задания;

– обеспечивает доступ пользователя к результатам расчетов;

– ведет учет потребленных пользователями вычислительных ресурсов.

Функционирование СУЗ сопровождается рядом внешних и внутренних событий. К внешним событиям относятся поступление задания в очередь, преждевременное завершение задания, удаление задания из очереди или прерывание выполнения задания пользователем или администратором, старт и останов СУЗ, изменение числа доступных вычислительных модулей (например, вывод из решающего поля по причине отказа, добавление в решающее поле новых или отремонтированных

модулей), смена режимов планирования. Режимом планирования определяются различные настроечные параметры и ограничения СУЗ, например, приоритеты пользователей, системные таймауты, максимально или минимально допустимые размер задания, число заданий в очереди и другие. К внутренним событиям можно отнести запуск очередного задания на выполнение в назначенное планировщиком время в соответствии с построенным расписанием. О времени и типе каждого события СУЗ делает отметки в своем журнале учета событий.

При моделировании поведения СУЗ на вход модели необходимо подать модельный поток внешних событий, который модель СУЗ должна обрабатывать, генерируя выходной поток внутренних событий. Неформальное определение полностью адекватной модели СУЗ в этом случае можно сформулировать следующим образом. На одном и том же потоке внешних событий реальная СУЗ и ее полностью адекватные модели должны генерировать идентичные потоки внутренних событий. Другими словами, расписания, построенные планировщиками полностью адекватных моделей СУЗ, должны точно совпадать с расписанием реальной СУЗ при одном и том же входном потоке внешних событий.

Отметим сразу, что событийная модель СУЗ предполагает независимость поступающих внешних событий, что в реальных системах выполняется не всегда. Например, если СУЗ ограничивает число заданий в очереди от одного пользователя, то при достижении этого предела некоторым пользователем будет проявляться следующая зависимость: чем раньше задание этого пользователя завершится, тем раньше от него поступит в очередь новое задание. Другим примером может служить удаление пользователем задания из очереди в случае, если прогнозируемое время старта превышает для пользователя срок актуальности его задания. Для простоты в настоящей работе пренебрежем этими фактами и будем считать внешние события независимыми.

Под моделированием СУЗ суперкомпьютера будем понимать процесс подачи на вход модели потока внешних событий и фиксацию потока внутренних событий. При этом адекватность модели, определяемую степенью схожести результирующего потока внутренних событий с аналогичным потоком реальной системы, будем называть адекватностью в широком смысле. Для составления прогнозов запусков заданий необходимо иметь адекватную в широком смысле модель СУЗ.

Очевидно, что эта модель будет иметь идентичные с реальной системой статистические показатели, рассчитываемые за интервал времени: загрузку (утилизацию) решающего поля суперкомпьютера, среднее время ожидания задания в очереди, среднюю длину очереди и другие. При исследовании влияния изменений параметров СУЗ или характеристик входного потока заданий на статистические интервальные показатели (средние значения и дисперсии показателей за определенный интервал времени), как правило, не выдвигаются требования к точному совпадению выходных потоков внутренних событий. В этом случае модель СУЗ будет адекватной, если разница каждого из измеряемых показателей входного и выходного модельных потоков не превышает дисперсии этого показателя в стационарном режиме. Например, пусть время ожидания задания в очереди для реальной системы составляет 60 минут со среднеквадратичным отклонением в 10 минут. Тогда можно назвать адекватной модель СУЗ, которая на том же входном потоке событий на выходе позволяет получить среднее время ожидания задания в очереди 63 минуты с дисперсией в 5 минут. Модель, обеспечивающую на одном и том же входном потоке внешних событий идентичность реальной системе интервальных статистических показателей, будем называть адекватной в узком смысле.

Существующие способы моделирования СУЗ [8–10] можно свести к следующим вариантам:

- построение аналитической модели СУЗ;
- натурный эксперимент на реальном суперкомпьютере;
- исследование СУЗ с виртуальным вычислителем;
- построение имитационной модели СУЗ.

Аналитическую модель СУЗ можно попытаться построить, руководствуясь теорией массового обслуживания, но при этом разработчик модели может столкнуться с рядом существенных проблем [11]. Математический аппарат хорошо проработан для простейших потоков (стационарных, ординарных и без последействия). Теория массового обслуживания оперирует в терминах интенсивности поступления заданий входного потока, производительности обслуживающих устройств и интервальных характеристик системы, таких как среднее время

ожидания в очереди. Аналитическая модель позволяет исследовать влияние изменений СУЗ на ее интервальные показатели, но не предоставляет механизм предсказания времени запуска отдельных заданий, необходимый для прогнозирования. В силу сложности построения и ориентации на интервальные показатели аналитическая модель не будет рассматриваться в настоящей работе.

Натурный эксперимент как метод моделирования СУЗ

Под натурным экспериментом будем понимать воспроизведение входного потока внешних событий на реально работающем суперкомпьютере. По определению, модель СУЗ в натурном эксперименте будет полностью адекватной. Тем не менее, покажем, что натурный эксперимент не может обеспечить воспроизведение результатов моделирования (то есть потока внутренних событий) со 100-процентной точностью. Время обработки прошедшего очередь задания складывается из трех, в общем случае случайных, величин:

– времени запуска задания: времени, затрачиваемого СУЗ на выделение вычислительных модулей и их конфигурацию в соответствии с требованиями задания;

– времени непосредственного выполнения задания на выделенных ВМ;

– времени завершения задания: времени, затрачиваемого СУЗ на освобождение выделенных ВМ, в том числе на контроль завершения всех процессов задания, удаление созданных заданием временных файлов и разделяемых ресурсов, переконфигурацию ВМ и т.п.

Будем называть время запуска и завершения задания накладными расходами, которые большинством СУЗ в биллинговой подсистеме не отделяются от времени выполнения задания. При этом доля накладных расходов является случайной величиной и может зависеть от многих факторов, таких как сетевые задержки, изменение состояния вычислений в ядре операционной системы и т.п. Рассмотрим экспериментальные данные по измерению накладных расходов, полученные путем многократного запуска задания с известным и неизменным временем выполнения. Эксперименты производились на суперкомпьютере МВС-10П ОП (раздел Broadwell), установленном в *Межведомственном суперкомпьютерном центре РАН* (МСЦ РАН). Каждый запуск производился 5 раз, результаты усреднялись и рассчитывалось среднеквадратическое отклонение. В таблице 1 показана зависимость среднего времени обработки от числа ВМ для задания со временем выполнения 60 секунд.

Таблица 1

Время выполнения задания со временем обработки 60 секунд

Table 1

Task execution time with a processing time of 60 seconds

Число ВМ	Средние накладные расходы, с	Среднеквадратическое отклонение, с
1	9,7	1,2
2	7,3	2,9
3	10,3	0,6
4	10,7	0,6
5	10,7	1,2
20	15,0	6,2

Из таблицы можно сделать ошибочный вывод, что накладные расходы на запуск и останов задания растут с числом ВМ. Детальное рассмотрение серии экспериментов для 20 ВМ показало, что накладные расходы составили для разных запусков трижды 11 секунд, один раз 17 секунд и один раз 25 секунд. Значение в 25 секунд можно интерпретировать как «выброс», связанный со сложностью одновременного выделения 20 ВМ для задания. С ростом числа ВМ можно говорить о возрастании дисперсии накладных расходов. Изменение времени обработки задания (1 минута, 3 минуты, 10 минут) не оказало влияния на средние накладные расходы, которые сохранились на отметке в 10 секунд. Время запуска задания в эксперименте в среднем составляет 30 % от накладных расходов, оставшиеся 70 % приходятся на завершение задания. Отметим, что в эксперименте время выполнения задания было искусственно сделано постоянным, тогда как из-за взаимного влияния каналов передачи данных и иных факторов время выполнения одного и того же реального задания при разных запусках может отличаться.

Главным недостатком натурного эксперимента можно назвать трудность его проведения с организационно-экономической точки зрения, поскольку дорогостоящие суперкомпьютерные ресурсы в ходе подобного эксперимента будут дублировать уже проведенные расчеты. Если подобный эксперимент и можно будет провести, то только в течение сравнительно небольшого временного промежутка. При этом для получения достоверных статистических данных требуется сбор информации

о работе системы в течение дней, недель и месяцев. На практике натурный эксперимент применяется путем изменения исследуемых параметров СУЗ в процессе эксплуатации суперкомпьютера и дальнейшего анализа влияния внесенных изменений на показатели качества СУЗ. По изменению этих показателей принимается решение о сохранении изменений или о возврате к предыдущей версии настроек СУЗ.

Моделирование СУЗ с виртуальным вычислителем

Для моделирования поведения СУЗ может быть использован виртуальный вычислитель – программная подсистема, которая вместо запуска заданий на ВМ решающего поля делает пометку о том, что ВМ заняты на время выполнения задания. Реальные вычисления в этом случае не производятся. Время выполнения задания обычно определяется модельным потоком внешних событий в зависимости от способа его генерации: например, на основе анализа статистики работы реальной системы либо заданного распределения.

Существуют два способа моделирования СУЗ с виртуальным вычислителем: в режиме реального времени и в режиме модельного времени. В реальном времени виртуальный вычислитель представляется для СУЗ в качестве решающего поля, СУЗ фактически работает в режиме натурного эксперимента без запуска заданий на суперкомпьютере. Это позволяет говорить о точности такого моделирования как сопоставимой с точностью натурного эксперимента. Недостатком такого способа является длительное время моделирования, соответствующее реальному времени работы СУЗ. Тем не менее, высокая точность и возможность не занимать в эксперименте реальные ресурсы суперкомпьютера делают этот способ предпочтительным в случаях, когда не требуется срочное получение результатов моделирования.

В основе моделирования СУЗ с виртуальным вычислителем в режиме модельного времени лежит идея о «продвижении» системного времени в те моменты, когда внешних или внутренних событий не происходит. Для примера: если в некоторый момент времени эксперимента новых заданий не поступает, в текущий момент одно задание обрабатывается и оно завершится через час, то существует возможность продвинуть системное время на час вперед. Моделирование в этом случае существенно ускоряется. В работе [12] недельный эксперимент был проведен за 3 дня с использованием модельного времени. Для реализации этого способа требуется разработка специального программного средства продвижения системного времени с дополнительной верификацией точности результатов эксперимента. Заметим, что откат модельного времени при таком способе моделирования невозможен, поскольку не дает построить прогнозную модель СУЗ для случаев, когда внешние события поступают в модель в реальном времени.

Имитационное моделирование СУЗ

Любая имитационная модель СУЗ будет представлять собой некоторый независимый объект, который в заданных условиях по поведению аналогичен исследуемой СУЗ. Независимая от самой СУЗ реализация имитационной модели имеет ряд преимуществ и недостатков. Выделим существенные преимущества имитационных моделей:

– модель может быть создана до реализации СУЗ с целью определения целесообразности реализации СУЗ в целом;

– модель может быть создана для СУЗ с закрытым исходным кодом;

– построенная модель позволяет получить результаты эксперимента в короткие сроки (секунды и минуты для анализа месяцев работы системы).

К недостаткам имитационного моделирования можно отнести:

– длительное время построения имитационной модели;

– потребность в изучении средства моделирования;

– необходимость валидации разработанной модели, то есть определения точности воспроизведения моделируемой системы.

Основой большинства средств построения имитационных моделей является событийная система, меняющая состояния участвующих объектов в дискретные моменты времени. Распространение получили два способа отсчета времени: периодический опрос по таймеру и переход к следующему событию [13]. При опросе по таймеру модельное время сдвигается на заданную величину, после чего обрабатываются все события, время наступления которых прошло за этот период. В результате этого способа продвижения времени наступление события соблюдается с точностью до периода сдвига. Для реализации второго способа от-

счета времени требуется хранить время наступления всех событий в системе. Затраты на хранение этих временных меток позволяют повысить точность наступления событий.

Существующие средства построения имитационных моделей для систем управления заданиями можно разделить на три класса: языки моделирования, программные платформы, симуляторы.

Языки моделирования полностью обеспечивают процесс моделирования – продвижение модельного времени и взаимодействие объектов в системе, позволяя исследователю сосредоточиться на описании существенных свойств и характеристик имитационной модели. При этом исследователь должен самостоятельно воспроизвести в модели всю логику работы СУЗ: сформировать вычислитель с заданными характеристиками и порядок обработки заданий в вычислителе, создать планировщик заданий с заданным алгоритмом действий, описать источники поступления входного потока заданий, разработать программные модули для проведения эксперимента и сбора нужных выходных данных. Для построения имитационной модели СУЗ могут быть использованы такие специализированные языки, как AnyLogic [14], ExtendSIM [14, 15], GPSS World [16], Simulink [14, 17].

AnyLogic является языком моделирования общего назначения, разрабатывается он с 2000 года (последний релиз в 2018 году). Разработка моделей производится в графическом интерфейсе, поддерживается использование языка программирования Java для доработки компонент. ExtendSIM разрабатывается с 1987 года (последний релиз в 2019 году). ExtendSim включает большое число библиотек имитационного моделирования, ориентированных на различные предметные области. ExtendSim не требует специальных знаний и навыков программирования, имеет дружественный к пользователю интерфейс, для моделирования достаточно нарисовать структурную схему моделируемого процесса и с помощью настроек необходимых параметров блоков ввести исходные данные. В качестве языка расширения используется внутренний язык программирования ModL. Одним из ранних языков моделирования был GPSS World, созданный в 1961 году (последний релиз в 2018 году). Вся разработка модели ведется на языке GPSS. Программа на языке GPSS представляет собой последовательность операторов, отображающих происходящие события. Simulink является языком моделирования, разрабатываемым с 1984 года (последний релиз в 2018 году). Simulink предоставляет тесную интеграцию с MATLAB.

Программные платформы имитационного моделирования СУЗ позволяют сократить время на реализацию модели за счет реализованных в платформе частей моделируемых систем и компонентов для отображения различных данных, например, статистических. Программная платформа предоставляет типовые сущности, такие как «вычислительный модуль», «задание», «планировщик заданий» с широким набором различных характеристик. Разработчик формирует свою модель из готовых крупных блоков и конфигурирует их под решаемую задачу. Распространены такие программные платформы, как SimGrid [18], GridSim [19], GSSIM [20].

SimGrid – это программная платформа для разработки симуляторов распределенных приложений, разрабатываемая с 1999 года (последний релиз в 2019 году). SimGrid предоставляет разработчику модели две основные сущности: ресурсы и задачи. Под ресурсом понимаются как вычислительный ресурс, так и сетевой канал передачи данных. Задачи характеризуются стоимостью и текущим состоянием. GridSim широко используется различными исследователями для моделирования грид-систем и СУЗ [13]. Разрабатывается с 2002 года, последний релиз в 2017 году. GSSIM разрабатывается с 2007 года, поддерживает использование языка программирования Java для доработки компонентов.

Симуляторы СУЗ предоставляют разработчику готовую к исследованию модель СУЗ, которую необходимо сконфигурировать. Выделим наиболее общие шаги по конфигурированию симулятора СУЗ:

– настроить модель ресурсов (какие ресурсы требуются заданию, из каких ресурсов и какой конфигурации состоит вычислитель);

– настроить источник данных для формирования входного потока (определить параметры его конфигурации или задать входной файл определенного вида);

– настроить алгоритм планирования, выбрав один из существующих или реализовав собственный алгоритм планирования;

– настроить формат выходных данных для отображения результата.

Некоторые из указанных шагов могут потребовать разработки программного кода. Примеры симуляторов СУЗ: MONARC [21], Alea [22], OptorSim [23], WorkflowSim [24]. Для

указанных симуляторов доработка моделей ведется на языке программирования Java. MONARC разрабатывается с 2000 года и предназначен для анализа систем большого размера. Ключевым аспектом этого симулятора являются широкие возможности по мониторингу компонентов системы [25]. Alea основан на GridSim и разрабатывается с 2007 года. Основная задача Alea – исследование алгоритмов планирования, ряд которых уже реализован в симуляторе. WorkflowSim разрабатывается с 2012 года. Основное предназначение WorkflowSim состоит в оптимизации потока выполнения заданий [25]. OptorSim разрабатывается с 2003 года. В OptorSim существует возможность конфигурирования сетевой топологии между вычислительными модулями с заданием их пропускной способности и размера данных для задания.

Следует упомянуть также об эмуляторах СУЗ, таких как MicroGrid [26]. Отличительной чертой эмулятора является возможность совместного использования в эксперименте компонентов реальной системы и эмулируемых частей СУЗ, что позволяет перейти к полунатурному моделированию [27].

Существующие средства имитационного моделирования позволяют построить прогнозную модель СУЗ и проводить с ней эксперименты на любом входном модельном потоке событий. Однако, как уже отмечалось, для имитационных моделей необходимо производить валидацию результатов экспериментов с целью определения адекватности модели как в узком смысле, так и в широком. Для этого нужно задать меру адекватности, выразить эту меру некоторой количественной характеристикой и определить допустимые пределы значений этой характеристики, в границах которых модель будет считаться адекватной.

Событийная модель СУЗ

Пусть все события в СУЗ происходят в дискретные моменты времени t_i. Рассмотрим поток независимых заданий $J_1, J_2, \ldots J_k, \ldots$, поступающий на обработку в суперкомпьютерную систему, решающее поле которой состоит из m вычислительных модулей. Каждое поступающее в очередь задание J_i имеет следующие характеристики:

– момент поступления задания в систему r_i;

– требуемое количество вычислительных ресурсов p_i (в простейшем случае – необходимое число ВМ);

– заказанное время обработки e_i;

– реальное время обработки w_i, $0 \le w_i \le e_i$, которое складывается из времени запуска задания a_i, времени выполнения задания b_i и времени завершения задания c_i.

Обратим внимание, что реальное время обработки недоступно для системы управления заданиями и не может быть использовано для построения расписания. Как было показано выше, a_i, b_i и c_i являются случайными величинами и могут изменяться от запуска к запуску одного и того же задания.

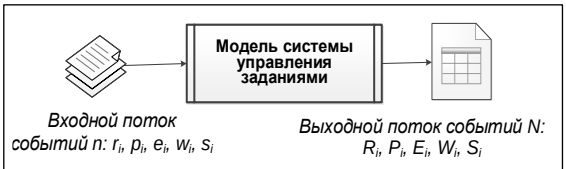
Планировщик определяет момент начала обработки задания s_i. Производными характеристиками задания являются:

– время ожидания задания в очереди $q_i = s_i - r_i$;

– время пребывания в системе $f_i = q_i + w_i$;

– момент завершения задания $g_i = s_i + w_i$.

В общем случае моделирование СУЗ представляет собой процесс, показанный на рисунке. На вход модели СУЗ поступает поток событий с некоторыми характеристиками. Результатом работы модели СУЗ является выходной модельный поток событий с другим набором характеристик. Обозначим характеристики этого потока заглавными буквами.



Событийная модель СУЗ

An event model of a supercomputer job management system

Известны три устоявшихся подхода к формированию входного потока событий [10, 28]. Первый подход состоит в использовании журнала учета событий реальной СУЗ. Подход позволяет воспроизвести входной поток событий реальной суперкомпьютерной системы с учетом всех ее особенностей.

Второй подход базируется на существующем стандарте записи журнала заданий SWF (Standard Workload Format) [29], в котором публикуются журналы событий некоторых суперкомпьютеров, в том числе университетских. Использование журналов сторонних СУЗ позволяет провести исследование для широкого круга разных суперкомпьютерных систем с различными характеристиками входных потоков заданий. Использование публичных SWF-

журналов позволяет также воспроизвести эксперименты других исследователей. Существенным минусом является неполнота потока событий: в SWF отражены только события, связанные с продвижением заданий в очереди, и отсутствует информация об изменении характеристик решающего поля (изменении числа ВМ), удалениях заданий из очереди или прерываниях выполнения заданий пользователем.

Третий подход заключается в генерации входного потока событий [30]. Каждый параметр задания (время поступления, заказанное и реальное время выполнения, требуемые вычислительные ресурсы) представляет собой случайную величину с определенным законом распределения. Закон и параметры распределения подбираются, как правило, на основе анализа журналов событий исследуемых суперкомпьютерных систем. Этот подход позволяет формировать несколько разных экземпляров входных потоков с одинаковыми распределениями.

Адекватность модели СУЗ

Предлагаемый авторами вариант определения меры адекватности в широком смысле основан на описанном в работе [27] способе оценки достоверности модели – валидации событий (event validity), когда сравниваются потоки событий моделируемой и реальной систем. В указанной работе не приводятся численные показатели, позволяющие сравнить два потока событий.

Определим меру близости двух потоков событий следующим образом. Сформулируем критерий недостоверности прогнозной модели. Модель является недостоверной, если число событий при моделировании не совпало с числом событий в реальной системе. Если какие-либо из событий не были воспроизведены при моделировании или возникли новые события, то модель недостоверна. Будем также считать модель недостоверной при несовпадении времени поступления задания в очередь в модели и реальной системе, при несовпадении заказанного времени выполнения задания или заказанных вычислительных ресурсов. Таким образом, модель является недостоверной, если $n \neq N$, $r_i \neq R_i$, $p_i \neq P_i$ или $e_i \neq E_i$.

Для полностью достоверной модели в эксперименте и в реальной системе совпадают число, порядок и время наступления всех событий. На практике построение полностью достоверной прогнозной модели СУЗ фактически невозможно даже для натурного эксперимента, как было показано выше.

Назовем диапазон между недостоверной и полностью достоверной моделями линейкой достоверности. Очевидно, все исследуемые адекватные в широком смысле модели СУЗ будут располагаться на этой линейке. Необходимо определить некоторый показатель, который будет служить мерой разности результатов моделирования на разных моделях и для каждой модели определять ее место на линейке достоверности.

При исследовании адекватности в узком смысле можно использовать математический аппарат проверки статистических гипотез [31]. В этом случае выдвигается статистическая гипотеза – предположение, что события входного и выходного потоков для некоторого уровня значимости принадлежат одному и тому же распределению случайной величины. Модель является адекватной в узком смысле, если события входного и выходного потоков являются случайными величинами, принадлежащими распределению одной и той же случайной величины.

Распространенным способом анализа зависимостей является корреляционный анализ. Корреляция отражает статистическую взаимосвязь случайных величин, но не определяет требуемую численную разницу двух потоков событий. Кроме того, математическое описание события включает множество несвязанных характеристик, которые нужно анализировать в совокупности. Необходимо разработать такое описание процесса моделирования, которое позволит упростить анализ. Подходящим результирующим описанием процесса моделирования может быть вектор значений, построенный как некоторый «слепок» всех характеристик события. Рассмотрим предлагаемый авторами вариант перехода от множества событий с различными характеристиками к одному вектору значений, позволяющему описать процесс моделирования.

Рассмотрим два модельных потока событий, представленных заданиями $j = (j_1, j_2, \ldots, j_n)$ и $J = (J_1, J_2, \ldots, J_N)$. Характеристики задания j_i: r_i (время поступления задания в очередь), p_i (требуемый объем ресурсов), e_i (требуемое время обработки), w_i (реальное время обработки), s_i (время запуска задания). Аналогичными характеристиками обладает задание $J_i = R_i, P_i, E_i, W_i, S_i$. Мера разности будет не определена в случае, если в потоках не совпа-

дают либо число заданий $n \neq N$, либо время поступления в систему какого-либо задания $r_i \neq R_i$, либо заказанные объемы ресурсов и время обработки для какого-либо задания $p_i \neq P_i$, $e_i \neq E_i$. В этой связи характеристики можно переписать следующим образом: $J_i = r_i$, p_i, e_i, W_i, S_i.

Построим два вектора размерности $n = N$. Для потока j определим вектор времени пребывания заданий в системе $v = (v_1, v_2, \ldots, v_n)$, $i \in (1, \ldots, n)$, где каждая компонента соответствует номеру задания в порядке его поступления в систему. Значение компоненты $v_i = (s_i - r_i + w_i)$ задается как время пребывания задания в системе, то есть сумма времени ожидания задания в очереди и времени его обработки. Для потока J аналогично определим вектор $V = (V_1, V_2, \ldots, V_n)$, $V_i = (S_i - R_i + W_i)$, $i \in (1, \ldots, n)$.

Таким образом, получены два вектора – v и V, различие между компонентами которых фактически определяет различие между двумя моделями СУЗ. Естественной мерой близости двух n-мерных векторов является евклидово расстояние между ними:

$$E = \sqrt{\sum_{i=1}^{n} (V_i - v_i)^2} . \quad (1)$$

Проверим независимость евклидова расстояния от длительности модельного эксперимента, измеренного в числе обработанных заданий. Сформируем некоторый вектор v потока j реальной системы, и на его основе сгенерируем вектор V модельного потока J. При генерации вектора V с вероятностью 50 % внесем изменения в 1 секунду в каждую компоненту вектора, то есть с вероятностью 0,5 значение времени обработки v_i i-го задания меняется на 1 секунду. При постоянных частоте изменений и средней величине изменения мера разности потоков не должна изменяться. При выполнении этого требования мера разности для короткого эксперимента будет совпадать с мерой близости для длительного эксперимента. Как показано в таблице 2, мера (1) зависит от продолжительности процесса моделирования, что делает неприменимым евклидово расстояние в качестве меры адекватности. Проведем нормализацию меры (1) и получим меру разности P потоков j и J:

$$P = \sqrt{\frac{\sum_{i=1}^{n} (V_i - v_i)^2}{n}} . \quad (2)$$

Как показывает таблица 2, мера (2) в отличие от евклидова расстояния не зависит от длительности модельного эксперимента при заданных величине и вероятности изменений.

Таблица 2

Мера разности для эксперимента с изменениями в 1 секунду для 50 % заданий

Table 2

Difference measure for the experiment with 1 second changes for 50 % of jobs

Число заданий	Мера разности *P*	Мера разности *E*	Число отличающихся заданий
50	0,71	5,0	25
100	0,77	7,8	60
250	0,74	11,7	138
500	0,71	15,9	253
750	0,72	19,8	392
1000	0,69	21,9	481
Коэффициент вариации	**0,039**	**0,49**	

Теперь при генерации модельного вектора V внесем изменение в 100 секунд в каждую компоненту с вероятностью 50 %. Результаты представлены в таблице 3, коэффициент вариации для меры разности равен 0,014.

Таблица 3

Мера разности для эксперимента с изменениями в 100 секунд для 50 % заданий

Table 3

Difference measure for the experiment with 100 second changes for 50 % of jobs

Число заданий	Мера разности *P*	Число отличающихся заданий
100	72,11	52
250	72,66	132
500	70,29	247
750	70,52	373
1 000	71,27	508

На основании таблиц 2 и 3 можно сделать вывод, что при сохранении средней величины изменения и частоты изменений мера разности P (2) не растет с увеличением числа заданий в сравниваемых экспериментах. Этот факт позволяет использовать меру разности P для любых по длительности модельных экспериментов в качестве меры адекватности модели.

Методика определения и эталонное значение меры адекватности модели СУЗ

Предлагается следующая методика определения меры адекватности. На основе анализа

статистики работы реальной суперкомпьютерной системы за достаточно длительный период определяется поток j, а на его основе – вектор v. Из потока j исключаются события s_i, связанные с моментами запуска заданий (внутренние события планировщика). Выделенный подпоток внешних событий подается на вход модели СУЗ, и в результате моделирования формируются поток J и соответствующий ему вектор V. Далее вычисляется мера разности потоков P. Чем меньше величина P, тем более адекватна модель СУЗ.

При $P = 0$ модель СУЗ будет полностью достоверной. Возникает вопрос о максимально допустимом значении меры $P^{\max}$, таком, что модель с мерой адекватности $P \le P^{\max}$ будет считаться адекватной.

Как было показано выше, повторение натурного эксперимента не дает точного воспроизведения результата. В то же время, поскольку реальная СУЗ адекватна самой себе, некоторая мера разности P^{ideal} потоков j и J, полученных в ходе двух повторений одного и того же натурного эксперимента, по определению будет меньше допустимого предела адекватности: $P^{ideal} \le P^{\max}$. Соответственно, любая модель с мерой разности $P \le P^{ideal}$ будет адекватной в широком смысле. Назовем P^{ideal} эталонным значением меры адекватности. Любая модель, имеющая меру адекватности, меньше эталона либо равную ему, не отличается по своему поведению от реальной системы.

Поскольку по перечисленным причинам проведение двух идентичных натурных экспериментов на практике сильно затруднено, предлагается определять эталонное значение меры адекватности путем сравнения результатов моделирования СУЗ с виртуальным вычислителем. На основе обработки статистики суперкомпьютера МВС-10П ОП, установленного в МСЦ РАН, авторами был сформирован модельный поток из 1 000 заданий, с которым было проведено моделирование отечественной СУППЗ с виртуальным вычислителем. Путем сравнения результатов моделирования с реальной системой были определены меры адекватности модели СУППЗ с виртуальным вычислителем для разного числа заданий. Результаты представлены в таблице 4. Столбец «Число заданий» соответствует числу k первых заданий потока j (реальная СУППЗ) и потока J (СУППЗ с виртуальным вычислителем). В столбце «Число отличающихся заданий» указано число заданий, для которых время ожидания в очереди в потоках j и J отличалось.

Таблица 4

Меры разности потоков заданий для модели СУППЗ с виртуальным вычислителем

Table 4

Difference measure of job streams for the SUPPZ with virtual nodes

Число заданий	Мера разности потоков	Число отличающихся заданий
50	0	0
100	12,0	4
250	11,4	13
500	12,0	20
750	11,6	28
1000	11,2	35

Из данной таблицы можно сделать вывод, что эталонное значение меры адекватности модели СУЗ, рассчитанной по формуле (2), равняется 12.

Заключение

Сложность современных СУЗ суперкомпьютеров возрастает, что обусловливает актуальность задачи моделирования СУЗ для определения ее оптимальных характеристик и прогнозирования времени запусков заданий. Авторами рассмотрены несколько методов моделирования СУЗ, среди которых выделены натурный эксперимент и моделирование СУЗ с виртуальным вычислителем. Показано, что результаты натурных экспериментов с одинаковыми условиями на одном и том же потоке заданий будут различаться. Очевидно, что при переходе к имитационному или аналитическому моделированию, существенно ускоряющему проведение экспериментов, степень различия с реальной системой возрастет, что поднимает вопрос об адекватности используемой модели СУЗ.

В настоящей статье предпринята попытка определить меру адекватности модели СУЗ через сравнение характеристик двух потоков заданий, один из которых получен от реальной суперкомпьютерной системы, а другой – от модели СУЗ. Каждое задание в потоках связывается с набором событий – поступлением в очередь, запуском на выполнение, завершением. Все события потока заданий авторы свели к единственному вектору, в котором каждая компонента соответствует определенному заданию и содержит время пребывания этого задания в системе. В статье показано, что мерой адекватности модели СУЗ в этом случае может служить рассчитываемое по формуле (2) нор-

мированное евклидово расстояние между векторами времен пребывания заданий в системе, полученными из потоков заданий реальной системы и модели СУЗ. На примере используемой в МСЦ РАН отечественной СУППЗ определено эталонное значение меры адекватности, соответствующее модели СУЗ с виртуальным вычислителем.

Работа выполнена в МСЦ РАН в рамках госзадания по проведению фундаментальных научных исследований, тема № 0065-2019-0016.

Литература

1. Reuther A., Arcand W., Bergeron B., Jones M., Prout A., Kepner J., Bestor D., Hubbell M., Michaleas P., Rosa A. Scalable system scheduling for HPC and big data. J. Parallel Distrib. Comput., 2018, vol. 111, pp. 76–92. DOI: 10.1016/j.jpdc.2017.06.009.

2. Yoo A.B., Jette M.A., Grondona M. SLURM: Simple Linux Utility for Resource Management. Job Scheduling Strategies for Parallel. Proc. JSSPP. LNCS, 2003, vol. 2862, pp. 44–60. DOI: 10.1007/10968987_3.

3. Henderson R.L. Job scheduling under the Portable Batch System. Job Scheduling Strategies for Parallel Proc. JSSPP. LNCS, 1995, vol. 949, pp. 279–294. DOI: 10.1007/3-540-60153-8_34.

4. СУППЗ. URL: http://suppz.jscc.ru/ (дата обращения: 12.07.2019).

5. Баранов А.В., Ляховец Д.С. Сравнение качества планирования заданий в системах пакетной обработки SLURM и СУППЗ. Научный сервис в сети Интернет: все грани параллелизма: сб. тр. Междунар. конф. М., 2013. С. 410–414. URL: http://agora.guru.ru/abrau2013/pdf/410.pdf (дата обращения: 12.07.2019).

6. Шабанов Б.М., Овсянников А.П., Баранов А.В., Лещев С.А., Долгов Б.В., Дербышев Д.Ю. Проект распределенной сети суперкомпьютерных центров коллективного пользования // Программные системы: теория и приложения. 2017. № 4. С. 245–262. DOI: 10.25209/2079-3316-2017-8-4-245-262.

7. Кузюрин Н.Н., Грушин Д.А., Фомин А. Проблемы двумерной упаковки и задачи оптимизации в распределенных вычислительных системах // Тр. ИСП РАН. 2014. № 1. С. 483–502.

8. Simakov N.A., Innus M.D., Jones M.D., DeLeon R.L., White J.P., Gallo S.M., Patra A.K., Furlani T.R. A Slurm Simulator: Implementation and Parametric Analysis. High Performance Computing Systems. Performance Modeling, Benchmarking, and Simulation. Proc. PMBS. LNCS, 2018, vol. 10724, pp. 197–217. DOI: 10.1007/978-3-319-72971-8_10.

9. Гергель В.П., Полежаев П.Н. Исследование алгоритмов планирования параллельных задач для кластерных вычислительных систем с помощью симулятора // Вестн. ННГУ. 2010. № 5. С. 201–208.

10. Феоктистов А.Г., Корсуков А.С., Дядькин Ю.А. Инструментальные средства имитационного моделирования предметно-ориентированных распределенных вычислительных систем // Системы управления, связи и безопасности. 2016. № 4. С. 30–60.

11. Баранов А.В., Ляховец Д.С. Влияние пакетирования на эффективность планирования параллельных заданий // Программные системы: теория и приложения. 2017. № 8. С. 193–208. DOI: 10.25209/2079-3316-2017-8-1-193-208.

12. Баранов А.В., Киселев Е.А., Ляховец Д.С. Квазипланировщик для использования простаивающих вычислительных модулей многопроцессорной вычислительной системы под управлением СУППЗ // Вестн. ЮУрГУ. 2014. Т. 3. № 4. С. 75–84. DOI: 10.14529/cmse140405.

13. Prajapati H.B., Shah V.A. Analysis perspective views of grid simulation tools, Journ. Grid Computing, 2015, vol. 13, no. 2, pp. 177–213. DOI: 10.1007/s10723-015-9328-9.

14. Yakimov I.M., Trusfus M.V., Mokshin V.V. and Kirpichnikov A.P. AnyLogic, ExtendSim and Simulink Overview Comparison of Structural and Simulation Modelling Systems. Proc. 3rd RPC, Vladivostok, 2018, pp. 1–5. DOI: 10.1109/RPC.2018.8482152.

15. Krahl D. ExtendSim 7. Proc. Simulation Conf., Miami, USA, 2008, pp. 215–221. DOI: 10.1109/WSC.2008.4736070.

16. Cox S.W. GPSS World: A brief preview. Proc. Simulation Conf., Phoenix, USA, 1991, pp. 59–61. DOI: 10.1109/WSC.1991.185591.

17. Giordano A.A., Levesque A.H. Getting Started with Simulink. In Modeling of Digital Communication Systems Using Simulink. John Wiley & Sons, 2015, 408 p. DOI: 10.1002/9781119009511.ch1.

18. Legrand A., Quinson M., Casanova H., Fujiwara K. The SimGrid project simulation and deployment of distributed applications. Proc. 15th IEEE Intern. Conf. on High Performance Distributed Computing, Paris, 2006, pp. 385–386. DOI: 10.1109/HPDC.2006.1652196.

19. Chelladurai S.R. Gridsim: A flexible simulator for grid integration study. MSc Thes., 2017. DOI: 10.24124/2017/1375.

20. Bak S., Krystek M., Kurowski K., Oleksiak A., Piatek W., Waglarz J. GSSIM – A tool for distributed computing experiments, scientific programming. 2011, vol. 19, no. 4, pp. 231–251. DOI: 10.3233/SPR-2011-0332.

21. Legrand I.C., Newman H.B. The MONARC toolset for simulating large network-distributed processing systems. Proc. Simulation Conf., Orlando, USA, 2000, vol. 2, pp. 1794–1801. DOI: 10.1109/WSC.2000.899171.

22. Klusacek D., Rudova H. Alea 2: Job scheduling simulator. Proc. 3rd ICST, 2010. DOI: 10.4108/ICST.SIMUTOOLS2010.8722.

23. Bell W.H., Cameron D.G., Millar A.P., Capozza, L., Stockinger K., Zini F. Optorsim: a grid simulator for studying dynamic data replication strategies. IJHPCA, 2003, vol. 17, pp. 403–416. DOI: 10.1177/10943420030174005.

24. Chen W., Deelman E. WorkflowSim: a toolkit for simulating scientific workflows in distributed environments. Proc. IEEE. 8th Intern. Conf. E-Sci., Chicago, 2012, pp. 1–8. DOI: 10.1109/eScience.2012.6404430.

25. Taheri J., Zomaya A., Khan S. Grid simulation tools for job scheduling and data file replication. Wiley, 2013, pp. 777–797.

26. Xia H., Dail H., Casanova H., Chien A.A. The MicroGrid: using online simulation to predict application performance in diverse grid network environments. Proc. 2nd Intern. Workshop CLADE, Honolulu, USA, 2004, pp. 52–61. DOI: 10.1109/CLADE.2004.130909.

27. Глинский Б.М., Родионов А.С., Марченко М.А., Подкорытов Д.И., Винс Д.В. Агентно-ориентированный подход к имитационному моделированию суперЭВМ экзафлопсной производительности в приложении к распределенному статистическому моделированию // Вестн. ЮУрГУ. 2012. № 18. С. 93–106.

28. Cirne W., Berman F. A model for moldable supercomputer jobs. Proc. 15th IPDPS, San Francisco, USA, 2001, p. 8. DOI: 10.1109/IPDPS.2001.925004.

29. Standard Workload Format. URL: http://www.cs.huji.ac.il/labs/parallel/workload/swf.html (дата обращения: 12.07.2019).

30. Lublin U., Feitelson D.G. The workload on parallel supercomputers: modeling the characteristics of rigid jobs. J. of Parallel and Distributed Computing, 2003, vol. 63, iss. 11, pp. 1105–1122. DOI: 10.1016/S0743-7315(03)00108-4.

31. Sargent R.G. Verification and validation of simulation models. Proc. Winter Simulation Conf., USA, 1994, pp. 77–87. DOI: 10.1109/WSC.1994.717077.

Software & Systems
DOI: 10.15827/0236-235X.128.581-594

Received 18.07.19
2019, vol. 32, no. 4, pp. 581–594

Methods and tools for modeling supercomputer job management system

A.V. Baranov [1], *Ph.D. (Engineering Sciences), Associate Professor, Leading Researcher, antbar@mail.ru, abaranov@jscc.ru*
D.S. Lyakhovets [2], *Research Associate, anetto@inbox.ru*

[1] *Joint Supercomputer Center of the Russian Academy of Sciences – branch of Federal State Institution Scientific Research Institute for System Analysis of the Russian Academy of Sciences, Moscow, 119334, Russian Federation*
[2] *Research & Development Institute "Kvant", Moscow, 125438, Russian Federation*

Abstract. The paper discusses the methods and tools of modeling supercomputer job management systems, such as SLURM, PBS, Moab, and the domestic management system of parallel job passing. There are highlighted job management system modeling methods including modeling with real supercomputer system, JMS modeling by a virtual nodes, and simulation modeling. The authors consider methods and tools for constructing a model job stream.

The management system of parallel job passing example shows the impossibility of accurate reproducing a full-scale experiment with real supercomputer. The paper investigates the adequacy of the job management systems model in a broad and narrow sense. It is shown that an adequate in the narrow sense job management system model ensures compliance only with interval indicators and cannot be used as a forecast model. The authors consider a numerical estimate of the proximity of two event streams in order to determine the adequacy in a broad sense. The first event stream is the stream of real supercomputer events. The second one is the stream

of events produced by a job management systems model. The normalized Euclidean distance between two vectors corresponding to the compared streams is proposed as a measure of proximity of two streams. The vectors' dimension is equal to the number of processed jobs, the vectors components are the job residence times in the job management systems.

The method of adequacy determination is based on a comparison of the real supercomputer statistics and the results of job management systems modeling. The adequacy measure reference value is determined as the normalized Euclidean distance between the vectors of job residence times in the real system and in the job management system model.

Keywords: high performance computing, grid, job management system, supercomputer job scheduling, simulation, model adequacy.

Acknowledgements. *The work has been done in JSCC RAS in the framework of the fundamental research state task, topic no. 0065-2019-0016.*

References

1. Reuther A., Arcand W., Bergeron B., Jones M., Prout A., Kepner J., Bestor D., Hubbell M., Michaleas P., Rosa A. Scalable system scheduling for HPC and big data. *J. of Parallel and Distributed Computing*. 2018, vol. 111, 76–92. DOI: 10.1016/j.jpdc.2017.06.009.

2. Yoo A.B., Jette M.A., Grondona M. SLURM: Simple Linux Utility for Resource Management. *JSSPP 2003. LNCS*. Berlin, Heidelberg, Springer Publ., 2003, vol. 2862, pp. 44–60. DOI: 10.1007/10968987_3.

3. Henderson R.L. Job scheduling under the Portable Batch System. *JSSPP 1995. LNCS*. Berlin, Heidelberg, Springer Publ., 1995, vol. 949, pp. 279–294. DOI: 10.1007/3-540-60153-8_34.

4. *SUPPZ*. Available at: http://suppz.jscc.ru/ (accessed July 12, 2019) (in Russ.).

5. Baranov A.V., Lyakhovets D.S. Comparison of the quality of job scheduling in workload management systems SLURM and SUPPZ. *Scientific Services & Internet: All Facets of Parallelism: Proc. Intern. Supercomputing Conf*. Novorossiysk, Russia, 2013, pp. 410–414. Available at: http://agora.guru.ru/abrau2013/pdf/410.pdf (accessed July 12, 2019) (in Russ.).

6. Shabanov B., Ovsiannikov A., Baranov A., Leshchev S., Dolgov B., Derbyshev D. The distributed network of the supercomputer centers for collaborative research. *Program Systems: Theory and Applications*. 2017, vol. 8, no. 4, pp. 245–262. DOI: 10.25209/2079-3316-2017-8-4-245-262 (in Russ.).

7. Kuzyurin N.N., Grushin D.A., Fomin S.A. Two-dimensional packing problems and optimization in distributed computing systems. *Proc. ISP RAS*. 2014, vol. 26, no. 1, pp. 483–502 (in Russ.).

8. Simakov N.A., Innus M.D., Jones M.D., DeLeon R.L., White J.P., Gallo S.M., Patra A.K., Furlani T.R. A Slurm Simulator: Implementation and Parametric Analysis. *High Performance Computing Systems. Performance Modeling, Benchmarking, and Simulation. PMBS 2017. LNCS*. vol. 10724, pp. 197–217. DOI: 10.1007/978-3-319-72971-8_10.

9. Gergel V.P., Polezhaev P.N. The study of parallel job scheduling algorithms for cluster computing systems using a simulator. *Vestn. of Lobachevsky Univ. of N. Novgorod*. 2010, no. 5, pp. 201–208 (in Russ.).

10. Feoktistov A.G., Korsukov A.S., Dyadkin Yu.A. Toolkits for simulation modeling of subject-oriented distributed computing systems. *Systems of Control, Communication and Security*. 2016, no. 4, pp. 30–60 (in Russ.).

11. Baranov A., Lyakhovets D. The influence of packaging on efficiency of parallel jobs scheduling. *Program Systems: Theory and Applications*. 2017, vol. 8, no. 1, pp. 193–208. DOI: 10.25209/2079-3316-2017-8-1-193-208 (in Russ.).

12. Baranov A.V., Kiselev E.A., Lyakhovets D.S. The quasi scheduler for utilization of multiprocessing computing system's idle resources under control of the management system of the parallel jobs. *Bulletin of the South Ural State Univ., Series Computational Mathematics and Software Engineering*. 2014, vol. 3, no. 4, pp. 75–84. DOI: 10.14529/cmse140405 (in Russ.).

13. Prajapati H.B., Shah V.A. Analysis perspective views of grid simulation tools. *J. Grid Computing*. 2015, vol. 13, no. 2, pp. 177–213. DOI: 10.1007/s10723-015-9328-9.

14. Yakimov I.M., Trusfus M.V., Mokshin V.V. and Kirpichnikov A.P. AnyLogic, ExtendSim and Simulink overview comparison of structural and simulation modeling systems. *Proc. 3rd Russ.-Pacific Conf. on Computer Technology and Applications (RPC)*. Vladivostok, 2018, pp. 1–5. DOI: 10.1109/RPC.2018.8482152.

15. Krahl D. ExtendSim 7. *2008 Winter Simulation Conf*. Miami, FL, USA, 2008, pp. 215–221. DOI: 10.1109/WSC.2008.4736070.

16. Cox S.W. GPSS World: A brief preview. *Proc. 1991 Winter Simulation Conf*. Phoenix, AZ, USA, 1991, pp. 59–61. DOI: 10.1109/WSC.1991.185591.

17. Giordano A.A., Levesque A.H. Getting started with Simulink. *Modeling of Digital Communication Systems Using Simulink*. 2015. DOI: 10.1002/9781119009511.ch1.

18. Legrand A., Quinson M., Casanova H., Fujiwara K. The SimGrid project simulation and deployment of distributed applications. *15th IEEE Intern. Conf. on High Performance Distributed Computing*. Paris, 2006, pp. 385–386. DOI: 10.1109/HPDC.2006.1652196.

19. Chelladurai S.R. *Gridsim: A Flexible Simulator for Grid Integration Study*. MSc Thes., 2017. DOI: 10.24124/2017/1375.

20. Bak S., Krystek M., Kurowski K., Oleksiak A., Piatek W., Waglarz J. GSSIM – A tool for distributed computing experiments. *Scientific Programming*. 2011, vol. 19, no. 4, pp. 231–251. DOI: 10.3233/SPR-2011-0332.

21. Legrand I.C., Newman H.B. The MONARC toolset for simulating large network-distributed processing systems. *Winter Simulation Conf. Proc*. Orlando, USA, 2000, pp. 1794–1801, vol. 2. DOI: 10.1109/WSC.2000.899171.

22. Klusacek D., Rudova H. Alea 2: job scheduling simulator. *Proc. 3rd ICST*. 2010. DOI: 10.4108/ICST.SIMUTOOLS2010.8722.

23. Bell W.H., Cameron D.G., Millar A. P., Capozza L., Stockinger K., Zini F. Optorsim: A grid simulator for studying dynamic data replication strategies. *IJHPCA*. 2003, vol. 17, pp. 403–416. DOI: 10.1177/10943420030174005.

24. Chen W., Deelman E. WorkflowSim: A toolkit for simulating scientific workflows in distributed environments. *2012 IEEE 8th Intern. Conf. on E-Sci*. Chicago, IL, 2012, pp. 1–8. DOI: 10.1109/eScience.2012.6404430.

25. Taheri J., Zomaya A., Khan S. Grid simulation tools for job scheduling and data file replication. *Scalable Computing and Communications: Theory and Practice*. Wiley Publ., 2013, pp. 777–797.

26. Xia H., Dail H., Casanova H., Chien A.A. The MicroGrid: using online simulation to predict application performance in diverse grid network environments. *Proc. 2nd Intern. Workshop CLADE 2004*. 2004, Honolulu, HI, USA, 2004, pp. 52–61. DOI: 10.1109/CLADE.2004.130909.

27. Glinsky B.M., Rodionov A.S., Marchenko M.A., Podkorytov D.I., Weins D.V. Agent-oriented approach to simulate exaflop supercomputer with application to distributed stochastic simulation. *Bulletin of the South Ural State Univ., Series Mathematical Modeling, Programming & Computer Software*. 2012, no. 18, pp. 93–106 (in Russ.).

28. Cirne W., Berman F. A model for moldable supercomputer jobs. *Proc. 15th IPDPS 2001*. San Francisco, CA, USA, 2001, p. 8. DOI: 10.1109/IPDPS.2001.925004.

29. *Standard Workload Format*. Available at: http://www.cs.huji.ac.il/labs/parallel/workload/swf.html (accessed July 12, 2019).

30. Lublin U., Feitelson D.G. The workload on parallel supercomputers: modeling the characteristics of rigid jobs. *J. of Parallel and Distributed Computing*. 2003, vol. 63, iss. 11, pp. 1105–1122. DOI: 10.1016/S0743-7315(03)00108-4.

31. Sargent R.G. Verification and validation of simulation models. *Proc. Winter Simulation Conf*. USA, 1994, pp. 77–87. DOI: 10.1109/WSC.1994.717077.

Для цитирования

Баранов А.В., Ляховец Д.С. Методы и средства моделирования системы управления суперкомпьютерными заданиями // Программные продукты и системы. 2019. Т. 32. № 4. С. 581–594. DOI: 10.15827/0236-235X.128.581-594.

For citation

Baranov A.V., Lyakhovets D.S. Methods and tools for modeling supercomputer job management system. *Software & Systems*. 2019, vol. 32, no. 4, pp. 581–594 (in Russ.). DOI: 10.15827/0236-235X.128.581-594.