Методы и программное обеспечение анализа исторических данных суперкомпьютерных планировщиков

Петунин С. А.
ФГУП «ВНИИА» им. Н.Л. Духова
Москва, Россия
e-mail: petunin@vniia.ru

Воронцов А. Г.
ФГУП «ВНИИА» им. Н.Л. Духова
Москва, Россия
e-mail: voroncov-ag@narod.ru

Аннотация

В работе рассмотрен подход к построению технологии массового счета в условиях промышленных суперкомпьютерных центров, который базируется на следующих принципах:

- использования расширенной типизации приложений при запуске вычислительных заданий;
- создания реалистических моделей рабочей нагрузки;
- регулярном применении аппарата анализа рабочей нагрузки для оценки эффективности выполнения прикладных программ и управления процессом массового счета.

Описаны две методики для характеризации параметров рабочей нагрузки. Важным результатом проведенной работы стало внедрение принципа типизации приложений в практику проведения расчетов, а также разработка программного обеспечения анализа трассировочных файлов, формируемых планировщиком заданий.

1. Введение

Основная задача любого суперкомпьютерного центра (СКЦ) заключается в обеспечении проведения эффективных высокопроизводительных вычислений. Для достижения этой цели необходимо создание и использование адаптивной технологии взаимодействия технического персонала и пользователей, которая бы учитывала не только особенности аппаратуры вычислительных кластеров, но и специфику рабочей нагрузки запускаемых приложений. Построение подобной системы обработки информации, ориентированной на качественное проведение массовых расчетов, приводит к созданию и применению различных политик выделения и использования вычислительных ресурсов. При этом, организационные и технологические процессы управления эксплуатацией нуждаются в наличии необходимого уровня инструментария, который, в частности, мог бы обеспечить хорошее понимание характеристик и шаблонов рабочей нагрузки конкретного СКЦ.

Начиная с 2011г. во ФГУП «ВНИИА» ведется разработка и внедрение компонент системного программного обеспечения, направленных на улучшение процесса сопровождения вычислительных кластеров и повышения степени загрузки их ресурсов [1-3]. К текущим задачам эксплуатации, при решении которых используется системное ПО, обычно относятся: мониторинг ресурсов СКЦ, управление вычислительными заданиями пользователей, анализ статистических данных и поддержка обратной управляющей связи. Далее мы опишем подход к созданию кластерной модели вычислительных заданий и методики анализа при реализации аналитической компоненты рабочей нагрузки для промышленного СКЦ.

2. Классификационная модель рабочей нагрузки для промышленных СКЦ

Большинство моделей рабочей нагрузки опираются на следующую интерпретацию [5-6]:

1) определение совокупности вычислительных заданий как объектов потребления суперкомпьютерных ресурсов;
2) выделение групп субъектов-пользователей, генерирующих эти задания и имеющих определенную активность и различия поведения по запросу ресурсов.

В условиях промышленных СКЦ появляется возможность значительно расширить этот подход, в первую очередь, путем проведения классификации входного потока заданий на базе типов программных приложений. Любой расчет, проводимый на суперкомпьютерных мощностях промышленного предприятия, не является результатом запуска удаленным пользователем работы с неизвестным типом расчетного задания (как это обычно происходит в вычислительных центрах коллективного пользования), а связан с определенной тематикой и планом работы предприятия. Варианты многофакторной или иерархической классификации вычислительных работ для управления массовым счетом зависят от особенностей проведения вычислений. В качестве примера можно привести классификацию вычислительного задания, применяемую в ряде вычислительных центров нашей страны еще с досуперкомпьютерных времен:

1) проект (project) – название головной плановой темы предприятия, по которой

проводится расчет;

2) задача (task) – название задачи, объединяющей группу запусков заданий, направленных на получение результата по одной расчетной модели;
3) программный пакет (program) – название прикладной программы, по которой производится моделирование;
4) методика/научная область (area) – название области исследования/применения расчета;
5) и т.д.

Для обеспечения подобной классификации необходимо обязать пользователя указывать эти дополнительные поля при запуске задания. Это легко сделать, реализовав процедуру запроса паспортной формы задания, например, в прологе планировщика. Если применить аппарат кластеризации по нескольким типам-факторам к входному потоку заданий, то становится возможным сделать шаблоны рабочей нагрузки более разнообразными и, соответственно, более реалистическими. Таким образом, предлагается расширить традиционную модель кластеризации входного потока заданий на основе поведения пользователей [6] следующим образом:

$W = \{K^F_i\}$

i =1,2,…n (n – количество кластеров);

F= <project | task | user |…> (тип фактора)

Методика кластеризации: k-means

Параметры вектора: $<A, \{R_i\}, \{K_i\}>$

A – активность по фактору (процент заданий);

R_i – запрашиваемые ресурсы: (например, $R_1(nodes)$ – количество запрашиваемых узлов;

R_2 ($Tlim$) – запрашиваемое время на исполнение);

K_i – дополнительная группа параметров.

В список запрашиваемых ресурсов иногда включают запрос на оперативную память, но в большинстве сценариев запуска для заданий при монопольном заказе вычислительного узла этот параметр несущественен и опускается. В число дополнительных параметров, расширяющим кластерное пространство и характеризующих входной поток, можно включить коэффициент точности запроса времени исполнения задания K_{wall}. Для алгоритма backfill планировщика заданий эта характеристика является важной.

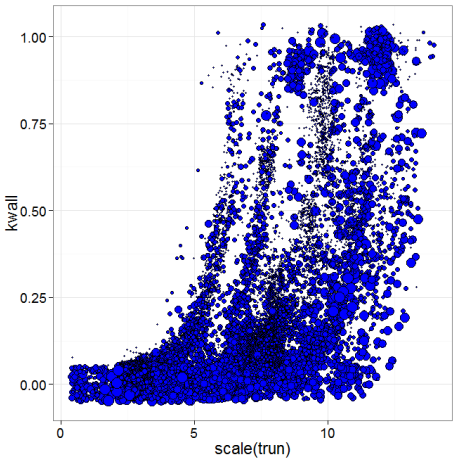


Рис.1. *Kwall* – коэффициент точности прогноза: отношение реального времени (*Trun*) выполнения задания к запрошенному времени (*Tlim*)

3. Методология формирования шаблонов поведения входного потока заданий

Применим следующую методику создания шаблонов входного потока.

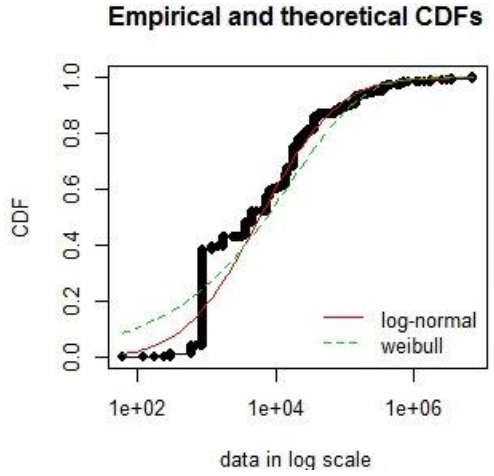
1. Согласно предложенной модели проведем кластеризация для каждого типа фактора F.
2. Для каждого параметра вектора кластеризации определим типовые метрики описательной статистики: среднее, медиана, стандартное отклонение и т.д.
3. По возможности, по эмпирическим распределению параметров, подберем соответствующие теоретические распределения.
4. Полученную статистическую модель можно использовать для исследования работы новых алгоритмов планировщика заданий в СКЦ, расширяя шаблоны нагрузки, моделирующие только поведение пользователей, тем самым обеспечивая симулятор более реалистическим профилем потока.

В табл.1 приведен пример вычисления статистических характеристик для параметра «запрос времени выполнения» при группировке входного потока в 3 кластера, проведенной для 6-ти месячной выборки.

Кластеры (фактор «task»)	**Доля заданий**	**mean T_{LIM} (min.)**	**sd T_{LIM} (min.)**
$K^{TASK}1$	70%	11	80
$K^{TASK}2$	1%	678	157
$K^{TASK}3$	29%	27	161

Табл.1. Характеристики параметра timelimit при кластеризации по фактору «task»

Рис.2 иллюстрирует подбор теоретической функции распределения для этого параметра в кластере К3.



Goodness-of-fit statistics		
	log-normal	weibull
Kolmogorov-Smirnov statistic	0.2099936	0.2068115
Cramer-von Mises statistic	137.5962831	187.5472524

Рис.2. Подгонка эмпирической CDF для T_{LIM} кластера *K3* теоретической CDF

Полученные статистические шаблоны поведения входного потока по каждому параметру в каждом кластере целесообразно использовать симуляторами рабочей нагрузки при выборе и тюнинге алгоритмов управления заданиями.

4 Анализ трассы рабочей нагрузки. Критерии эффективности

Существует три штатных подхода для анализа эффективности выполнения приложения:

1) проведение профилирования своей программы самим пользователем;
2) сбор и анализ некоторых данных о выполнении программы на уровне системы мониторинга;
3) анализ показателей эффективности использования ресурсов, сохраненных планировщиком в исторической базе данных.

Третий путь является наиболее простым решением, позволяя делать предварительные оценки эффективности выполнения заданий. Сделаем попытку проанализировать соответствие расчетных моделей со способностями прикладных пакетов к масштабированию. Будем опираться на анализ базы данных планировщика, содержащей записи с результатами запусков заданий. Применим следующую методику: в качестве индикаторов эффективности алгоритмов решателей программ определим и исследуем поведение двух параметров:

1. Доля затрат на системное время – коэффициент эффективности K_{ef1}. Этот показатель вводится как отношение процессорного времени, когда работали компоненты ядра операционной системы кластера, к времени суммарной реальной работы, включающей пользовательскую фазу задания. Реальное время содержит множитель аллокации CPU, который не влияет на вычисление коэффициента при операции деления.

$$K_{ef1} = T_{sys} / (T_{sys} + T_{user})$$

Если его величина, находящаяся в интервале (0,1), превышает значение 0.5 – это говорит о том, что более половины времени своего выполнения программа тратила на вызовы ядра операционной системы, занимаясь обменами или другими системными работами с использованием CPU.

2. Второй индикатор показывает отношение разницы между потребленным вычислительным ресурсом кластера (процессорным временем T_{CPU}) и суммой времен системной и пользовательской фазы задания, используемых в предыдущей формуле. Можно попытаться проинтерпретировать это отношение как коэффициент “непараллельности” задания – K_{ef2}.

$$K_{ef2} = (T_{CPU} - (T_{sys} + T_{user})) / T_{CPU}$$

Именно такая доля процессорного времени не была загружена пользовательским заданием.

Проблема использования столь простых, но мощных показателей, заключается в некотором недоверии к значениям T_{sys} и T_{user} , сохраняемых планировщиком slurm. Определено, что задания снимающиеся пользователем или по тайм-ауту, могут не передавать в планировщик для суммирования системное и пользовательское время со всех незаконченных процессов задания. Поэтому, для повышения достоверности коэффициентов при анализе необходимо отбирать только задания, имеющие успешный код завершения. Тем не менее, в общем случае в качестве индикаторов «тревожности» о неэффективном использовании вычислительных ресурсов конкретным приложением, безусловно, введенные коэффициенты использовать целесообразно.

Выберем некоторый календарный период эксплуатации СКЦ. Затем сгруппируем задания из log-файла рабочей нагрузки по названию программных пакетов (фактор «program»), выбрав топ из 5-ти наиболее часто запускаемых программ. Таблица со средними значениями введенных нами показателей приведена ниже.

Программа	Среднее K_{ef2}	Среднее K_{ef1}
P_11	0.78	0.42
P_2	0.40	0.06
P_9	0.21	0.04
P_16	0.20	0.06
P_7	0.04	0.01

Табл. 2. Коэффициенты эффективности программ

Высокие значения показателя неиспользования процессорного времени K_{ef2} у P_11 являются тревожным сигналом целесообразности запусков в многоузловом режиме. Ниже приводится график, иллюстрирующий размах значений данного

коэффициента для следующих 2-х прикладных пакетов (P_11 и P_9), который наглядно показывает изменение коэффициента K_{ef2} при масштабировании узлов. В качестве типа представления графических данных выберем компактный статистический boxplot («ящик с усами»).

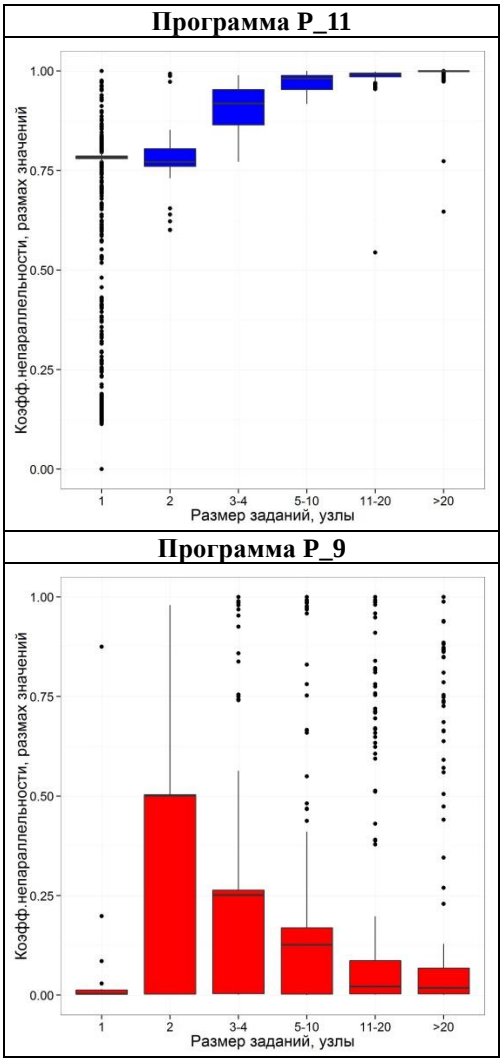


Рис. 3. Boxplot для значений K_{ef2} по группам узлов

Из рисунка можно сделать следующий вывод: P_11 и P_9 демонстрируют две противоположные зависимости от масштабирования узлов: в то время как у P_11 наблюдается экспоненциальное ухудшение коэффициента при переходе в группу с более ресурсоемкими заданиями, программа P_9 увеличивает утилизацию процессорного времени.

Поддержка описанных выше методик анализа эффективности реализована в составе общих возможностей аналитического программного комплекса ANTIK. Программа обеспечивает следующий основной функционал:

- проведение статистического анализа временных, числовых и факторных показателей выполненных параллельных программ, включая формирование описательных статистик, функций распределения для заданных метрик и выявления зависимостей между характеристиками;
- формирование отчетов с обобщенной статистикой по различным группам;
- визуализация всех статистических данных как базовых, так и консолидированных в виде различных графических представлений;
- обеспечение интерактивности работы аналитика с возможностью замены и масштабирования исследуемых показателей непосредственно на визуализируемых графиках;
- проведение анализа эффективности выполненных заданий с помощью специальных оценочных коэффициентов;
- унификацию и легкость импорта исторических данных планировщика slurm в систему анализа из любой суперкомпьютерной системы без необходимости описания ее вычислительной конфигурации.

5. Заключение

В данной работе были предложены методики для создания реалистических шаблонов моделирования рабочей нагрузки промышленных суперкомпьютерных центров, а также для вычисления параметров-индикаторов эффективности программных приложений на основе анализа трассировочных log-файлов.

Важным моментом проведенной работы стало создание системы статистического анализа, ориентированной на процесс поддержки эксплуатации СКЦ [4]. С помощью этого системного программного обеспечения были проведены исследования для данной статьи. Кроме того, это программное обеспечение стало рабочим инструментом для управления и регулирования массовым счетом на предприятии.

Список используемых источников

1. Novikov A.B, Petunin S.A., “Alternative job scheduling algorithms impact on computing cluster utilization in specific use cases”, Proc. XII International Seminar Supercomputations and mathematical modeling, Sarov: VNIIEF, 2011.

2. Ivanov K.V. “Monitoring system with forecasting errors”, Proc. International Conf. on Parallel Computational Technologies PAVT’2013, p.592 Chelyabinsk, 2013.

3. Petunin S.A., Ivanov K.V., Novikov A.B., Management of HPC Clusters: Development and Maintenance. // Proc. of the 15-th International Workshop on Computer Science and Information Technologies CSIT’2013’, Vienna-Budapest-Bratislava, pp.43-46, 2013.

4. Petunin S.A., “Методика начального анализа рабочей нагрузки вычислительных кластеров”. Труды Четвертой Международной научной конференции Информационные Технологии и Системы, Изд. Челябинского государственного университета, pp-129-130, Челябинск, 2015.

5. A. K. Mishra, J. L. Hellerstein, W.Cirne , C.R. Das, “Towards Characterizing Cloud Backend Workloads: Insights from Google Compute Clusters,” ACM SIGMETRICS Performance Evaluation Review, vol. 37, no. 4, pp. 34-41, March 2010.
6. I.S.Moreno, P.Garraghan, P.Townend, J.Xu, “Analysis, Modeling and Simulation of Workload Patterns in Large-Scale Utility Cloud”, IEEE Transactions on Emerging Topics in Computing, 2014.