УДК 004.046

ФУНКЦИОНАЛЬНО-ОРИЕНТИРОВАННАЯ ИЕРАРХИЧЕСКАЯ ВЫЧИСЛИТЕЛЬНАЯ СРЕДА

Я. М. Далингер

FUNCTION-ORIENTED HIERARCHICAL COMPUTATIONAL ENVIRONMENT

Ya. M. Dalinger

В статье представлены результаты анализа задачи формирования вычислительной среды, ориентированной на решение заданного набора прикладных задач, что соответствует ее функциональной ориентации для обслуживания конкретного предприятия.

Результаты работы могут быть полезны администраторам и разработчикам автоматизированных систем управления предприятиями.

The paper presents the problem of establishing a computational environment focused on the solution of a broad applied tasks list. The system is used to atomize the management procedures of a given company. The obtained results can be useful for both administrators and developers of the operation support systems.

***Ключевые слова**:* вычислительная среда, поток запросов, матрица интенсивностей, матрица логических связей, функционал.

Keywords: computational environment, flow of queries, intensity matrix, matrix of logical relations, functional.

Введение

Создание автоматизированных систем сбора и обработки информации для крупных предприятий, предусматривает создание соответствующей вычислительной среды, ориентированной на решение множества конкретных прикладных задач необходимых для осуществления предприятием заданного вида деятельности.

Здесь под вычислительной средой (Compute environment, Distributed Cjmputing Environment) понимается комплекс аппаратных и программных средств, обеспечивающих решение прикладных задач [8; 9]. Понятие вычислительной среды используется также в работе [2].

Поскольку предприятия занимаются различными видами деятельности, вычислительная среда должна быть функционально ориентированной под задачи предприятия, что подразумевает не только высокую эффективность решения задач, но и отсутствие избыточности, рациональную загрузку оборудования и каналов связи, достаточную функциональную надежность.

Как показывает практика, большинство известных вычислительных сред, реализуются в виде вычислительных сетей и имеют логическую иерархическую структуру, что объясняется соответствием взаимодействия решаемых задач, реальной структуре предприятия и взаимодействию процессов внутри него. Кроме того иерархическая структура (особенно логиическая) имеет такие положительные особенности как простота маршрутизации; простота декомпозиции задач обработки информации и управления для решения на различных уровнях; простота организации взаимодействия процессов на различных уровнях.

Однако, наряду с перечисленными достоинствами, у иерархических структур имеются и недостатки:

– высокая интенсивность потоков данных, поступающих на верхние уровни;

– наличие эффектов поглощения и тиражирования данных при их движении снизу вверх и сверху вниз;

– сильная зависимость функциональной надежности всей структуры от надежности верхних уровней.

В связи с этим представляет интерес исследование иерархических вычислительных сред и возможностей их функциональной ориентации для решения задач конкретного предприятия, а также постановка общей задачи построения иерархической вычислительной среды.

Описание среды

Как отмечалось, иерархическая вычислительная среда состоит из совокупности взаимодействующих программных и аппаратных компонентов, которые ориентированы на решение заданного набора функциональных прикладных задач. Прикладные задачи при их программной реализации разбиваются на множество взаимодействующих процессов, установленных и исполняемых на различных обслуживающих устройствах (серверы, микропроцессорные модули, рабочие станции и т. д.). Процессы взаимодействуют путем обмена запросами, которые содержат необходимые данные.

В общем случае иерархическая вычислительная среда задается следующими параметрами:

Число уровней иерархии – H ($1 \le H < \infty$), при этом, чем меньше номер уровня, тем уровень ниже.

Вектор числа обслуживающих устройств на каждом уровне иерархии – $\mathbf{n} = (n_1, n_2, ..., n_H)$, где $1 \le n_i \le \infty$ и $N = \sum_{i=1}^{H} n_i$ – общее число обслуживающих устройств в вычислительной среде.

Матрица логических связей между устройствами – $\mathbf{C} = \left\| c_{ij} \right\|$ (i, j = 1,2,…, N), где $c_{ij} = 1$, если требуется передача запросов от устройства i к устройству j, и $c_{ij} = 0$, если не требуется передача запросов от устройства i к устройству j. Порядковый номер обслуживающего устройства находящегося на уровне k, имеющего там номер m формируется по правилу: $i = m + \sum_{j=1}^{k-1} n_j$. В этом случае по заданному порядково-

му номеру устройства однозначно определяются его уровень и номер в уровне.

Правила обслуживания и формирования выходящих потоков запросов на каждом обслуживающем устройстве. Правила обслуживания поступающих на обработку запросов определяют длительность обслуживания запроса, дисциплину очереди запросов и формирование выходящего потока. Возможен случай, когда выходящий поток имеет интенсивность большую чем входящий (в результате обработки запроса на выходе обслуживающего устройства образуется несколько новых запросов) – эффект тиражирования запросов. Также возможен эффект поглощения, когда в результате обработки нескольких запросов (возможно разных потоков), на выходе обслуживающего устройства образуется один запрос [5].

Количество прикладных задач, решаемых в среде – M ($1 \le M < \infty$).

Вектор интенсивностей запуска задач на исполнение (интенсивностей потоков задач) – $\boldsymbol{\gamma} = (\gamma_1, \gamma_2, ..., \gamma_M)$, где $0 \le \gamma_j < \infty$ – интенсивность запуска на исполнение задачи j.

Вектор количества процессов, программно реализующих задачи $\mathbf{m} = (m_1, m_2, ..., m_M)$, где m_j – число процессов для реализации задачи j.

Множество матриц распределения процессов различных задач по обрабатывающим устройствам $\mathbf{S} = (\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_M)$, $\mathbf{S}_j = \left\| s_{jkm} \right\|$ (k = 1,2,…, m_j; m = 1,2,…,N) матрица распределения процессов задачи j по обслуживающим устройствам, где $s_{jkm} = 1$, если процесс номер k задачи j выполняется на обслуживающем устройстве номер m и $s_{jkm} = 0$, если процесс номер k задачи j не выполняется на обслуживающем устройстве номер m. Далее считаем, что для решения задачи ее процессы должны выполняться в порядке возрастания их номеров.

Множество матриц интенсивностей внешних потоков запросов, поступающих для исполнения процессов различных задач $\mathbf{A} = (\mathbf{A}_1, \mathbf{A}_2, ..., \mathbf{A}_M)$,

$$\mathbf{A}_j = \left\| \alpha_{jlm} \right\|, (l = 1,2,\ldots,L;\ m = 1,2,\ldots, m_j),$$

где $0 \le \alpha_{jlm} < \infty$ – интенсивность внешнего потока l, поступающего для исполнения процесса m задачи j.

Множество матриц средних значений длительностей исполнения процессов, на различных обслуживающих устройствах – $\mathbf{Z}_0 = (\mathbf{Z}_{01}, \mathbf{Z}_{02}, ..., \mathbf{Z}_{0M})$, $\mathbf{Z}_{0j} = \left\| z_{0jkm} \right\|$ (k = 1,2,…, m_j; m = 1, 2,…, N), где z_{0jkm} это средняя величина длительности исполнения процесса k задачи j на обслуживающем устройстве m, когда оно свободно от других процессов.

Множество матриц интенсивностей потоков запросов между процессами различных задач $\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_M)$, здесь $\mathbf{B}_j = \left\| \beta_{jin} \right\|$, ($i,\ n$ = 1,2,…, m_j) где $0 \le \beta_{jin} \le \infty$ – интенсивность потока запросов задачи j от процесса i задачи jк процессу n (число запросов, которые образует задача j от процесса i к процессу n).Можно считать, изменив определение задачи, что каждой задаче соответствует один процесс.

Вектор вероятностей отказов обслуживающих устройств $\mathbf{p}(t) = (p_1(t), p_2(t), ..., p_N(t))$, где $p_i(t)$ – вероятность отказа устройства i в момент t.

Общие результаты

Матрицы $\mathbf{C} = \left\| c_{ij} \right\|$, $(\mathbf{Z}_{01}, \mathbf{Z}_{02}, ..., \mathbf{Z}_{0M})$, $(\mathbf{S}_1, \mathbf{S}_2, ..., S_M)$, $(\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_M)$, $\mathbf{A}$, вектор $\boldsymbol{\gamma}$, а также правила обслуживания и формирования выходящих потоков запросов на каждом обслуживающем устройстве, дают возможность определить матрицы интенсивностей потоков запросов, поступающих на каждое обслуживающее устройство –

$$\boldsymbol{\Lambda}(\mathbf{C}, \mathbf{S}.\mathbf{B}) = \left\| \lambda_{ri} \right\| \ (r,\ i = 1,2,\ldots,\ N),$$

где $0 \le \lambda_{ri} < \infty$ – интенсивность потока запросов, поступающих на обслуживающее устройство i от обслуживающего устройства r.

Значение λ_{ri} вычисляется по формуле:

$$\lambda_{ri} = \sum_{j=1}^{M} \sum_{l=1}^{L} \sum_{m=1}^{m_j} a_{jlm} s_{jmi} + \sum_{j=1}^{M} \sum_{k=1}^{m_j} \sum_{r=1}^{m_j} \gamma_j \beta_{jkm} s_{jkr} s_{jmi} .$$

Здесь первое слагаемое – это суммарная интенсивность внешних потоков запросов, поступающих для исполнения процессов, установленных на обслуживающем устройстве i, а второе слагаемое – это суммарная интенсивность потоков запросов, поступающих от процессов, исполняемых на обслуживающем устройстве r, процессам, исполняемым на обслуживающем устройстве i.

Теоретическое значение средней длительности ожидания запросами в очереди к обслуживающему устройству i -W_i (i = 1,2,…, N) может вычисляться с применением математических моделей, учитывающих дисциплину очереди, производительность обслуживающего устройства и его загрузку, когда расчетная длительность исполнения процесса зависит от загрузки, правила обслуживания и формирования выходящих потоков, исследованных, например, в работах [4; 5; 6; 7].

Основной характеристикой качества работы вычислительной среды является время решения прикладных задач.

Существует нижний предел длительности решения задачи в среде, который можно получить, если рассматривать решение задачи в виртуальной среде, где нет других задач. Такую длительность для задачи j обозначим T_{0j}.

Теоретическое значение величины T_{0j} можно вычислить по формуле:

$$T_{0j} = \sum_{k=1}^{m_j} \sum_{m=1}^{N} s_{jkm} W_{0mj} + \sum_{k=1}^{m_j} \sum_{m=1}^{N} s_{jkm} z_{0jkm} .$$

Здесь W_{0mj} среднее время ожидания в очереди к обслуживающему устройству m запросами задачи j в виртуальной среде.

Поскольку в реальной системе задачи (процессы) могут конфликтовать за ресурсы обслуживающих устройств, расчетное время решения задачи j в реальной среде – T_j может отличаться от нижней границы, ($T_j > T_{0j}$) (j = 1,2,…,M). При расчете значения T_j имеем:

$$T_j = \sum_{k=1}^{m_j}\sum_{m=1}^{N} s_{jkm} W_{mj}(W_{0mj}, \rho_{jm}) + \\ + \sum_{k=1}^{m_j}\sum_{m=1}^{N} s_{jkm} z_{jkm}(z_{0jkm}, \rho_{jm}).$$

Здесь ρ_m – загрузка обслуживающего устройства номер m (понимается в смысле, определенном для систем массового обслуживания, например, в работе [4]), а среднее время ожидания в очереди к обслуживающему устройству m запросами задачи j – ($W_{mj}(W_{0mj}, \rho_{jm})$) и средняя длительность обработки запроса задачи j на обслуживающем устройстве m – $z_{jkm}(z_{0jkm}, \rho_{jm})$ от его загрузки обработкой запросов задачи j – $\rho_{jm} = \gamma_j \sum_{k=1}^{m_j}\sum_{i=1}^{m_j} s_{jkm}\beta_{jik} z_{0jkm}$ и значений аналогичных величин для виртуальной среды.

Общая загрузка обслуживающего устройства m зависит от суммарной загрузки исполнением запросов различных задач на этом устройстве – $\rho_m = \sum_{j=1}^{M} \rho_{jm}$.

Следует отметить, что при формировании среды необходимо обеспечить выполнение неравенства $\rho_m < 1$ для каждого m = 1,2,…, N [4]. Для этого требуется регулировать величину потоков каждой задачи j, поступающих на устройство m – Λ_{jm} (j = 1,2,…, m_j; m = 1,2,…, N) и размещение процессов по устройствам. Величину интенсивности достаточно просто вычислять для иерархических структур [1].

Загрузка может меняться, если меняется число устройств на соответствующем уровне. Если $q_{hn}(\mathbf{p}(t))$ вероятность того, что на уровне h ($1 \le h \le H$) в момент t работает n устройств ($1 \le n \le n_h$), то потоки запросов на работающие устройства распределяются по заданному правилу, которое учитывает возможность исполнения процессов на устройствах. Так, при равномерном распределении потоков получим: $\Pr(\Lambda_m) = q_{hn}(\mathbf{p}(t))\Lambda_{0h}/n$, где $\Pr(\Lambda_m)$ – вероятность того, что интенсивность суммарного потока запросов на обслуживающее устройство m равна Λ_m, Λ_{0h} – суммарная интенсивность потока запросов, поступающих на уровень h.

Таким образом, выше показано, как параметры вычислительной среды могут применяться для вычисления ее характеристик.

Задача формирования вычислительной среды

Определим варьируемые параметры иерархической вычислительной среды, которые можно менять в процессе настройки (функциональной ориентации) среды. К ним относятся: число уровней H, количество обслуживающих устройств на каждом уровне $\mathbf{n} = (n_1, n_2, ..., n_H)$, множество матриц распределения процессов различных задач по обрабатывающим устройствам $\mathbf{S} = (\mathbf{S}_1, \mathbf{S}_2, ..., \mathbf{S}_M)$.

Эффективность работы среды, естественно, зависит от длительности решения прикладных задач, которая зависит от качества программного обеспечения и организации работы среды. Анализ качества программного обеспечения в данной работе не рассматривается.

В качестве показателя эффективности организации вычислительной среды предлагается использовать величину:

$$E = \sum_{j=1}^{M} \frac{e_j}{1 + (T_j - T_{0j})} + \sum_{i=1}^{N} g_i \rho_i,$$

где e_j, g_i – весовые коэффициенты, определяющие важность задач и важность загрузки обслуживающих устройств.

Из формулы видно, что наибольшая эффективность достигается при равенстве реального времени решения задачи его нижней границе. Кроме того, показатель дает возможность учитывать вес (важность) каждой задачи с помощью коэффициентов $0 < e_j < \infty$, что требует эффективной организации среды для всех задач. Также показатель позволяет учитывать загрузку обслуживающих устройств, что важно при их высокой стоимости.

Организация работы среды характеризуется наличием очередей, загрузкой оборудования, вероятностью безотказной работы в течение установленного интервала времени. Эти характеристики влияют на длительность решения прикладных задач и, соответственно, значение показателя эффективности организации вычислительной среды.

В связи с этим целесообразно ввести вторичный показатель качества организации работы вычислительной среды, в виде функционала:

$$F(H, \text{n}, M, \text{m}, \text{C}, \text{B}, \text{S}, \text{A}, \text{p(t)}) = \\ = F_1(\Lambda\ (\text{C}, \text{S}, \text{B})) + F_2(H, \text{n}, \text{p(t)}).$$

В этой формуле

$F_1(\mathbf{\Lambda}(\mathbf{C}, \mathbf{S}, \mathbf{B})) = \sum_{i=1}^{N} a_i W_i + b_i(1 - \rho_i)$ – функционал, для вычисления величины затрат, связанных с ожиданием запросов в очередях к обслуживающим устройствам, и с простоем обслуживающих устройств. Здесь W_i, ρ_i – среднее время ожидания запросом в очереди к обслуживающему устройству номер i и загрузка обслуживающего устройства i, а

a_i, b_i – весовые коэффициенты, имеющие смысл затрат за единицу времени ожидания данными в очереди, и простоя обслуживающего устройства. Здесь время работы среды фиксируется, поскольку обычно задается длительность периодов работы среды до восстановления (профилактики) оборудования.

$$F_2(H, \mathbf{n}, \mathbf{p}(t)) = \sum_{i=1}^{N} g_i(\mathbf{p}(t)) + P(H, \mathbf{n}, \mathbf{p}(t)) \; -$$

функционал, определяющий величину затрат, связанных с отказами обслуживающих устройств и отказом всей структуры, где g_i – величина затрат, связанных с отказом обслуживающего устройства *i*, $P(H.\mathbf{n}, \mathbf{p}(t))$ – величина затрат, связанных с отказом среды (невозможностью выполнять заданные функции, решать заданный набор задач).

Длительность решения задачи складывается из длительности исполнения процессов на обслуживающих устройствах и длительности ожидания данными в очередях, а также длительности восстановления среды в случае отказа. Поэтому, минимизация функционала вторичного показателя качества организации среды, позволяет максимизировать общий показатель эффективности.

Выделение показателя эффективности и вторичного показателя качества дает возможность решать задачу функциональной ориентации среды путем декомпозиции и целенаправленного регулирования значений отдельных показателей для каждой задачи и каждого обслуживающего устройства.

Задача функциональной ориентации среды состоит в нахождении значений ее варьируемых параметров, при которых показатель эффективности организации среды работы имеет максимальное значение.

Предлагается следующая общая постановка задачи функциональной ориентации вычислительной среды:

Для заданных значений параметров среды: *M*, $\mathbf{m} = (m_1, m_2, ..., m_M)$, $\mathbf{A} = \|\alpha_{ij}\|$, $\mathbf{B} = (\mathbf{B}_1, \mathbf{B}_2, ..., \mathbf{B}_M)$, $\mathbf{Z}_0 = (\mathbf{Z}_{01}, \mathbf{Z}_{02}, ..., \mathbf{Z}_{0M})$, $\mathbf{p}(t) = (p_1(t), p_2(t), ..., p_N(t))$ найти $H_0, \mathbf{n}_0, \mathbf{S}_0$, при которых

$$E(H_0, \mathbf{n}_0, \mathbf{S}_0) = \max_{H, \mathbf{n}, \mathbf{S}} \{E =$$

$$= \sum_{j=1}^{M} \frac{e_j}{1 + (T_j - T_{0j})} + \sum_{i=1}^{N} g_i \rho_i \}.$$

В качестве ограничений возможно использовать следующие условия:

1. $F(H, \text{n}, M, \text{m}, \text{C}, \text{B}, \text{S}, \text{A}, \text{p(t)}) =$
$= F_1(\Lambda\ (\text{C}, \text{S}, \text{B})) + F_2(H, \text{n}, \text{p(t)}) < F_0$,

где F_0 – заданное по техническому заданию предельно допустимое значение функционала.

2. Для всех $j \sum_{k=1}^{m_j} \sum_{m=1}^{n} s_{jkm} s = m_j$ – все процессы задачи *j* распределены по обслуживающим устройствам.

3. Для всех *j* и всех $m \sum_{k=1}^{m_j} s_{jkm} \le m_j$ – на одном обслуживающем устройстве могут устанавливаться все процессы задачи *j*.

4. Для всех *j* и всех $k \sum_{m=1}^{N} s_{jkm} = 1$ – каждый процесс задачи *j* установлен на обслуживающем устройстве.

Возможны также ограничения, связанные с распределением процессов по уровням иерархической структуры, ограничения связанные с возможностями обслуживающих устройств.

Задачу можно рассматривать как задачу математического программирования, применяя для ее решения известные методы, например [3].

Следует отметить, что во многих случаях размерность среды (количество вычислительных устройств, количество задач) не очень велика (20, 30), в этом случае возможно решать задачу методом прямого перебора вариантов.

Выводы

Представленные результаты дают возможность формировать оптимальную структуру вычислительной среды, ориентированную на решение заданного набора прикладных задач. Формирование структуры вычислительной среды возможно проводить для оценки различных вариантов либо на этапе реконфигурации уже созданной структуры, когда известны параметры потоков данных, запросов и время обработки запросов, либо на этапе проектирования среды, применяя оценки величины необходимых параметров. Для иерархической структуры возможно получение зависимостей для вычисления значений функционала в достаточно простом для анализа и оптимизации виде.

Следует отметить, что в статье не учитывались особенности среды, связанные с каналами связи, что обусловлено высокой производительностью современных каналов связи и тем, что иерархическая структура является логической и может быть реализована как в сети Интернет, так и с использованием облачных технологий.

Возможным перспективным применением результатов может быть формирование требований к производительности вычислительных устройств при проведении замены или закупок оборудования, а также формирование требований к работе приложений (интенсивности потоков данных между приложениями, длительность исполнения и т. д.) для получения необходимых характеристик среды.

Литература

1. Анищенко, А. А. Вычисление нагрузок на узлы сети, представленной в виде дерева / А. А. Анищенко // Всероссийская конференция «Прикладная теория вероятностей и теоретическая информатика»: тез. докл. – М.: ИПИ РАН, 2012 – С. 49 – 51.

2. Артамонов, Г. Т. Топология регулярных вычислительных сетей и сред / Г. Т. Артамонов. – М.: Радио и связь, 1985. – 192 с.

3. Емеличев, В. А. Метод построения последовательности планов для решения задач дискретной математики / В. А. Емеличев, В. И. Комлик. – М.: Наука, 1981. – 208 с.

4. Вишневский, В. М. Теоретические основы проектирования компьютерных сетей / В. М. Вишневский. – М.: Техносфера, 2003. – 512 с.

5. Далингер, Я. М. Анализ потоков данных в системах с поглощением сообщений / Я. М. Далингер // Информатика и системы управления. – Комсомольск-на-Амуре: Изд-во Амурского гос. ун-та, 2012. – № 3(33). – С. 25 – 34.

6. Далингер, Я. М. Математические модели распределенной среды с поглощением и тиражированием сообщений / Я. М. Далингер. – СПб.: СПбГУ ГА, 2010. – 56 с.

7. Никитин, Е. В. Управление потоками данных в многосерверных системах обработки информации / Е. В. Никитин, Е. А. Саксонов // Информатика и системы управления. – Комсомольск-на-Амуре: Изд-во Амурского гос. ун-та, 2010. – № 3(25). – С. 3 – 9.

8. Першинов, В. И. Толковый словарь по информатике / В. И. Першинов, В. М. Савинков. – М.: Финансы и статистика, 1991. – 543 с.

9. Таненбаум, Э., М. ван Стеен. Распределенные системы. Принципы и парадигмы / Э. М. ванн Стеен Таненбаум. – СПб.: Питер, 2003. – 877 с.

Информация об авторе:

Далингер Яков Михайлович – кандидат технических наук, проректор по информатизации и региональному образованию Санкт-Петербургского государственного университета гражданской авиации, 8(812)704-15-82, iakovdalinger@gmail.com.

Yakov M. Dalinger – Candidate of Technical Science, Vice-President for IT and University Regional Branches, Saint-Petersburg State University of Civil Aviation.

Статья поступила в реколлегию 05.09.2013 г.