неопределенности с помощью современных вычислительных средств, в частности, p(arallel)-*matLab*. В последнее время появились принципиально новые задачи с т.н мягкими математическими моделями, которые в ряде случаев могут иметь весьма отдаленное отношение к реальной задаче. При этом решение задачи, м.б. даже очень низкой точности, должно быть безусловно получено. Эта идеология нашла свое развитие в том, что известный ПММ *MatLab* включил в свой состав новый комплекс программ, ориентированный на параллельные и распределенные вычисления в системе многоядерных процессоров - pMatlab toolbox. В связи с тем, что используются тензорные модели НП, автоматически возникает необходимость параллельной обработки данных. В работе авторами показаны способы, методы и алгоритмы параллельной обработки данных при решении задач в условиях неопределенности путем использования стандартного и специального математического обеспечения ПММ *MatLab*.

Матричный подход к параллельной обработке данных. Матрицы как математический объект широко используются для представления, в данном случае при помощи тензора моделируется НП [2]. Алгебра матриц хорошо разработана, основные положения теории матриц изложены в работах [3, 4], однако для понимания дальнейшего изложения рассмотрим некоторые аспекты алгебры матриц. Прямоугольная таблица чисел - матрица, сокращенное обозначение мат-рицы - $A = \{a_{ij}\}_{i=\overline{1,m}}^{j=\overline{1,n}}$. Для числовых матриц легко определяются привычные алгебраические операции сложения и умножения.

Важнейшей особенностью матриц является естественный параллелизм выполнения операций над ними. Для реализации операций над матрицами можем использовать систему из mxn процессоров, имеющих одновременный доступ к общей оперативной памяти, в которой расположены матрицы *A* и *B* [5]. Вычислительные системы подобные этой принято называть матричными. При этом связь каждого процессора со своими соседями обеспечивает применение конвейерного принципа обработки данных [1,3]. Развитие матричных моделей. Для того, чтобы элементы некоторого множества $\mathbb{S}$ могли быть элементами матрицы требуется наличие двух бинарных операций, замкнутых на этом множестве: аддитивной (+) и мультипликативной (*). Замкнутость операций означает, что если $a \in \mathbb{S}$ и $b \in \mathbb{S}$, то $(a+b) \in \mathbb{S}$ и $(a*b) \in \mathbb{S}$. Выбор аддитивной и мультипликативной операций может быть традиционным или определяться пользователем из условий задачи. Наиболее исследованы т.н., плоские матрицы, элементы которых можно трактовать как значения параметров реальных объектов, расположенных на плоскости в точках с целочисленными координатами. Однако на практике имеют место объекты, характеризующиеся большим числом целочисленных координат. Такие объекты можно рассматривать как точки пространства произвольного числа измерений (тензоры).

На основании анализа свойств многомерных матриц [5] сформулирован принцип последовательно-параллельного программирования, который позволяет упростить алгоритмизацию задач, для решения которых необходимо использование суперкомпьютеров.

Принцип последовательно - параллельного программирования определен в виде совокупности следующих утверждений [5]:

- в качестве базового типа данных используется алгебра многомерных матриц;
- операции этой алгебры реализуются как базовые процедуры обработки данных для конкретных суперкомпьютеров с максимально допустимой степенью параллелизма;
- многомерные матрицы включаются в языки программирования как самостоятельные объекты;

- алгоритмы представляются как последовательности операций над многомерными матрицами.

Основные достоинства принципа последовательно-параллельного программирования состоят в следующем:

- исключается связь между свойствами конкретной архитектуры суперкомпьютера и способностью данного алгоритма к распараллеливанию;

- ограничивается число реализуемых параллельно алгоритмов, можно наиболее полно использовать возможности суперкомпьютера для оптимизации программ, реализующих эти операции;

- алгоритм и программа, разрабатываются как обычные последовательные алгоритмы и программы, данные, представленные в виде многомерных матриц, последовательно обрабатываются операциями алгебры многомерных;

- программа выполняется на суперкомпьютере параллельно.

Постановки основных задач. Рассмотрим особенности параллельной обработки данных с точки зрения их конкретного применения для решения задач, в которых в качестве переменных выступают тензоры четных рангов [2]. ТП определена как:

$$T_x = x \otimes \mu_x^T .$$

Определение расстояния между ТП. Будем рассматривать матрицу как множество. Алгоритм определения расстояния между множествами (матрицами) *A* и *B* может быть таким. Находят для каждой строки и столбца всех матриц минимумы и максимумы, оценивают нижнюю p_{low} и верхнюю p_{high} границы расстояний между матрицами. Пусть $\rho(A, B)$ - расстояние между матрицами A_{nm} и B_{nm}, определяют матрицу $C=(c_{ij}|c_{ij} = ||A_{ij}-B_{ij}||)$. Очевидно, что $p_{low}(A, B) = \min(\sum_i \min_j C_{ij}, \sum_j \min_i C_{ij}) \le \rho(A,B) \le \max(\sum_i \max_j C_{ij}, \sum_j \max_i C_{ij})) = p_{high}(A,B)$.

По матрице *C* можно оценить границы $p_{low}(A, B)$ и $p_{high}(A, B)$, соответственно, проведя сравнение сразу с несколькими матрицами.

В работе расстояние между матрицами $\mathbf{A}= \left(a_{ij}\right)_{i=1,n}^{j=1,n}$ и $\mathbf{B}= \left(b_{ij}\right)_{i=1,n}^{j=1,n}$ предложено определять как: $d(\mathbf{A},\mathbf{B})= =\mathrm{sqrt}(\mathrm{trace}(\mathbf{A})^2-\mathrm{trace}(\mathbf{B})^2)$, для НП с треугольной ФП $\mathrm{trace}(\mathbf{A})= \lambda_1^A + \lambda_2^A + \lambda_3^A$ и $\mathrm{trace}(\mathbf{B})= \lambda_1^B + \lambda_2^B + \lambda_3^B$, λ_i^A, λ_i^B , i=1, 3 собственные значения матриц **A** и **B** соответственно. Отметим, что величина trace() является 1-м (и единственным) инвариантом тензора 2-го ранга, полученным как результат диадного произведения векторов.

Учитывая, что фадзификация ТП

$$T^x= \begin{pmatrix} 0 & 0 & 0 \\ x-\alpha x & x & x+\alpha x \\ 0 & 0 & 0 \end{pmatrix},$$

соответствующей $\tilde{x}$ с треугольной ФП, будет иметь вид: $\mathrm{trace}\,(T^x)=x$, можно утверждать, что в тензорной нотации расстояния между фадзифицированными и нечеткими объектами (на основании существующих правил определения нечеткой метрики) будут совпадать.

В нашем случае - $\tilde{A} = \{x_i / \mu_{\tilde{A}}(x_i)\}$, i=1, n; $\tilde{A} \to {}^A T_x = x_i \otimes \mu_{\tilde{A}}(x_i) = [t_{ij}]_{i=1,n}^{j=1,n}$, каждая ТП характеризуется следом $\mathrm{tr}({}^A T_x)= \sum_{i=1}^n \lambda_i$, из свойства следа как оператора вытекает, что четкая ТП, ближайшая к нечеткой ТП, будет расположена на наименьшем евклидовом расстоянии между следами, т.е. $[\mathrm{tr}({}^A T_x)- \mathrm{tr}({}^B T_x)]^2 \to \min$.

Следовые операции в системе тензор-переменных. Операции НМа в тензорном базисе связаны с операциями сверток, вычисления инвариантов и использования инвариантов как аналогов ТП. Показано [2], что ТП представляет собой диадный тензор, для которого существует только один ненулевой инвариант – след (не считая магнитуды-нормы), инварианты тензора могут быть определены как на основании элементов матрицы ТП, так и на основании собственных значений. Рассмотрим некоторые особенности 1-го инварианта.

В линейной алгебре [6], след n×n квадратной матрицы A определяется как сумма элементов главной диагонали A. Эквивалент, след матрицы – сумма ее собственных значений, что делает след инвариантом относительно изменения базиса. Этой характеристикой в общих чертах можно обычно определять след для линейного оператора. Отметим, след определен только для квадратной матрицы (т. е. n×n). Геометрически след может быть проинтерпретирован как бесконечно малое изменение в объеме (производная детерминанта), который точно определяется формулой *Jacobi's.*

Свойства. След – линейное отображение, таким образом $\mathrm{tr}(A+B)=\mathrm{tr}(A)+\mathrm{tr}(B)$, $\mathrm{tr}(cB)=c\cdot\mathrm{tr}(B)$ для всех квадратных матриц *A* и *B* и скаляра *c*. Если *A* - *m*×*n* матрица и *B* - *n*×*m* матрица, тогда tr(*AB*) = tr(*BA*). Названные свойства характеризуют след полностью в следующем смысле. Пусть *f* будет линейным функционалом в пространстве квадратных матриц, удовлетворяющих *f*(*xy*) = *f*(*yx*). Тогда *f* и tr - пропорциональны. След инвариантно подобен, что означает, что *A* и $P^{-1}AP$ имеют один и тот же след. Это вытекает из того, что

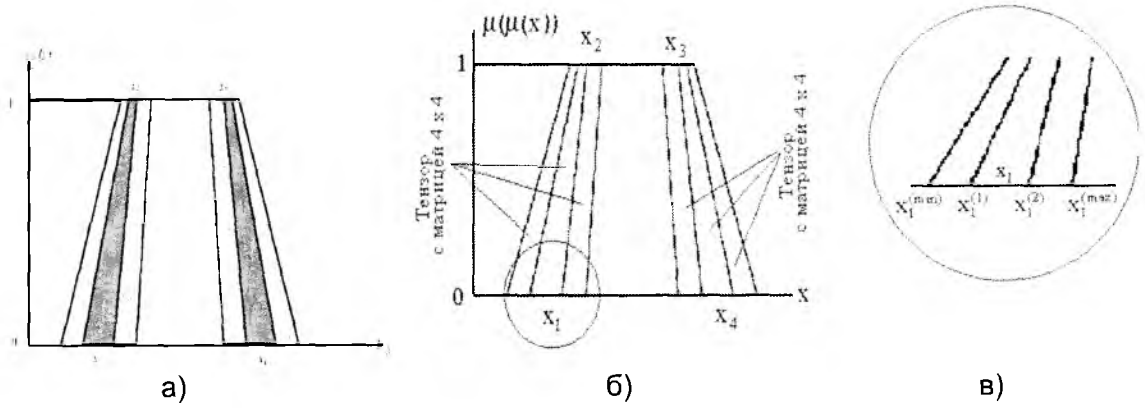
$$\mathrm{tr}(P^{-1}AP)=\mathrm{tr}((AP)P^{-1}=\mathrm{tr}(A).$$

Матрица и ее транспозиция имеют один и тот же след: $\mathrm{tr}(A)=\mathrm{tr}(A^T)$. Отметим, что порядок матриц влияет на значение следов: в общем случае $\mathrm{tr}(ABC)\neq\mathrm{tr}(ACB)$, однако след есть инвариантом для циклических перемещений, tr(*ABCD*) = tr(*BCDA*) = tr(*CDAB*) = =tr(*DABC*), т.е. две половины выражения можно только многократно переставить. След тензорного произведения 2-х матриц является произведением их следов:

$$\mathrm{tr}(x\otimes y)=\mathrm{tr}(x)\mathrm{tr}(y).$$

Отдельно рассмотрим *Kronecker'ово* произведение, необходимость в котором возникает при моделировании т.н. гипернечетких переменных (рис.6).

Определение [7]. Гипернечеткими множествами называются нечеткие множества, характеризующиеся функциями принадлежности трапецеидальной формы (нечетким интервалами), опорные точки которых в свою очередь сами являются нечеткими интервалами трапецеидальной формы.



а) б) в)

Рис.6-а - Представление гипернечеткого числа на плоскости; б), в) – параметры тензорных моделей гипернечеткого числа

Гипернечеткое число (переменная) можно представить как объединение ТП, каждое из которых представляет собой тензорное произведение $x_i^{(k)}\otimes\mu^T_{x_i^{(k)}}$, i=1, 4; $k\in\{^{(1),\ (2),\ (\min),\ (\max)}\}$, например, $T_x^{(1)}\Leftrightarrow$ $\{x_1^{(\min)}/\mu_{x_1^{(\min)}}, x_1^{(1)}/\mu_{x_1^{(1)}}, x_2^{(\min)}/\mu_{x_2^{(\min)}},$ $x_2^{(1)}/\mu_{x_2^{(1)}},\}$, $T_x=T_x^{(1)}\cup T_x^{(1)}\cup T_x^{(1)}\cup$

$\bigcup T_x^{(1)} \bigcup T_x^{(1)} \bigcup T_x^{(1)} \bigcup T_x^{(1)}$. В выражении для ТП компоненты могут быть матрицами, т.е. $[x_i^{(k)}] \otimes [\mu_{x_i^{(k)}}^T]$. В этом случае имеем Kronec-ker'ово произведением матриц, результат – матрица с блочной структурой, в которой каждый элементе первой матрицы заменяется элементом матрицы, масштабируемой этим элементом. Для матриц *U* и *V Kronecker'ово* произведение 2-х двумерных квадратных матриц может быть записано:

$$\begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} \otimes \begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} = \begin{bmatrix} a_{1,1}\begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} & a_{1,2}\begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} \\ a_{2,1}\begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} & a_{2,2}\begin{bmatrix} b_{1,1} & b_{1,2} \\ b_{2,1} & b_{2,2} \end{bmatrix} \end{bmatrix} = \begin{bmatrix} a_{1,1}b_{1,1} & a_{1,1}b_{1,2} & a_{1,2}b_{1,1} & a_{1,2}b_{1,2} \\ a_{1,1}b_{2,1} & a_{1,1}b_{2,2} & a_{1,2}b_{2,1} & a_{1,2}b_{2,2} \\ a_{2,1}b_{1,1} & a_{2,1}b_{1,2} & a_{2,2}b_{1,1} & a_{2,2}b_{1,2} \\ a_{2,1}b_{2,1} & a_{2,1}b_{2,2} & a_{2,2}b_{2,1} & a_{2,2}b_{2,2} \end{bmatrix}.$$

Результирующий ранг 4, результирующая размерность 16. Здесь ранг обозначает ранг тензора (количество необходимых индексов), в то время как ранг матрицы - число степеней свободы результирующего массива.

Параллельные вычисления в среде *pMatLab*.

Параллельный *MATLAB* [8, 9] позволяет реализовать многопроцессорную обработку в *MATLAB*, Тулбокс ПВ (*Parallel Computing Toolbox* - *PCT*) позволяет реализовать:

- прогон последовательной работы;
- прогон диалоговой (интерактивный режим) параллельной работы;
- прогон пакетной работы в *OSC* группе.

Параллельные вычисления в среде *MatLab* преследуют следующие цели:

- ускорение вычислений путем использования большого количества процессоров;
- использование большей памяти, чем это доступно в единственной машине.

Указанные цели реализованы следующим образом:

- использование *MPI* (*Message Passing Interface* - Интерфейс Прохождения, Сообщения), библиотеки, которые используются, чтобы заменить данные и управлять информацией между процессорами;
- работа в распределенных средах памяти;
- использование *OpenMP*: комплекта директив компилятора, который используется, чтобы выполнять нить (команд) параллельно в коллективной среде памяти. Отметим, что в реальных условиях параллельное программирование с использованием *C/C++/FORTRAN* и *MPI* чрезвычайно затруднено. Применение Параллельного *MATLAB* дает следующие преимущества: *MA-TLAB* широко используется для разработки/распространения алгоритмов, наличие языка высокого уровня (к тому же объектно-ориентированного) и встроенная среда Разработки/Визуализации ведет к продуктивной разработке кодов за достаточно короткое время программистами сравнительно невысокой квалификации.

Путем параллелизации *MATLA*B кода:

- алгоритм может выполняться с различными (в т.ч. большими) множествами данных;
- алгоритм может выполняться с большим размахом параметров;
- вычислительное время может быть уменьшено.

Отдельно отметим возможность прогона последовательной работы в среде *pMatLab*, т.к. в этом случае имитация параллельного выполнения программы дает возможность увидеть насколько она будет эффективной, если ее выполнять параллельно. Очень часто этот режим работ дает пользователю возможность избежать неоправданных затрат при создании параллельного кода.

Многопроцессорная обработка в *MATLAB. MATLAB R2008a* поддерживает виртуальную (мнимую) и явную мульти-процессорную обработку. Виртуальная мульти-процессорная обработка позволяет:

- обеспечить встроенность выполняемых программ в мультинить;
- ускорить выполнение многих процедур линейной алгебры, операций с матрицами;
- влиять при помощи достаточно большого числа ядер на процессор.

В свою очередь, явная мульти процессорная обработка позволяет обеспечить эффективные параллельные вычисления с использованием *PCT* и *MDSE* (*MATLAB Distributed Computing Server – MatLab* (сервер распределенной обработки) и влияет при помощи многочисленных процессоров на кластер. Явная мульти-процессорная обработка может быть реализована:

- только при помощи *PCT*;
- совместное использование *PCT* и *MDCS*

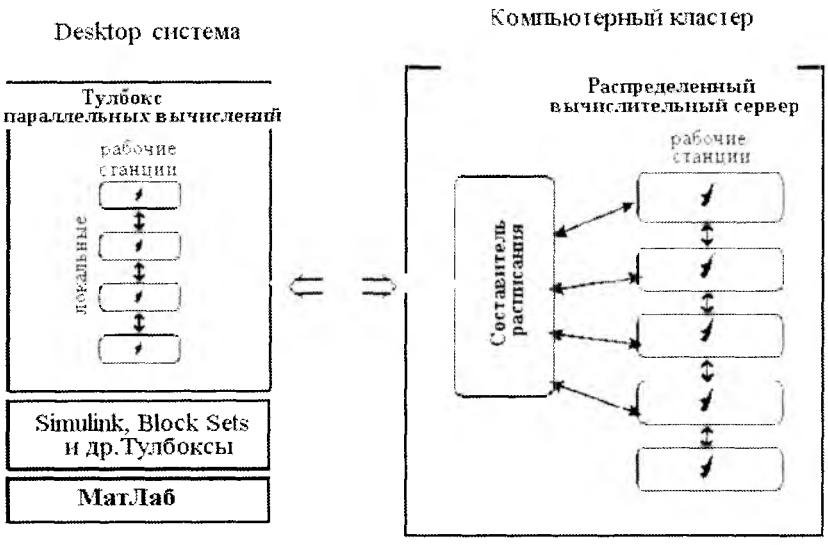


Рис.7. Явная мультипроцессорная обработка

Схема явной мультипроцессорной обработки приведена на рис.7. *PCT* обеспечивает параллельное создание на внутреннем языке *MATLAB*, например, параллельных циклов, распределенных массивов и сообщений и передачу сообщений. Позволяет быстрое макетирование параллельного кода через диалоговый параллельный *MATLAB* – сеанс и обеспечивает способность масштабировать проблему путем «впрягания» ресурсов в отдаленный кластер. Кроме того, расширения внутреннего языка включают *MATLAB* позволяют на стандартной технологии создавать и использовать распределенные массивы.

В состав расширений внутреннего языка *MATLAB* включены свыше 150 параллельных функций для использования на распределенных массивах - *cos, sin, log, find, isempty* и др. Особенно следует отметить наличие пакета *ScaLA-PACK*, включающего программы параллельной линейной алгебры - svd, lu и др. Кроме того, *PCT* поддерживает глобальные, коллективные операции как, например, глобальное сложение, редукция и др. и обеспечивает явный, хорошо структурированный параллелизм через *MPI* функции.

PCT поддерживает следующие типы планировщиков для представления работы:

- локальный планировщик, может одновременно работать с рабочими станциями, полезный для локальной отладки параллельного кода;
- менеджер работ, поддерживающий 3-х раздельный планировщик (*PBS, LSF, Microsoft CCS*);
- общий Планировщик;

- общий интерфейс, который допускает использование с 3-х раздельным планировщиком;

- дополнительно к *PCT* поддерживает использование конфигураций, которая, в свою очередь обеспечивает удобный путь, чтобы сохранить параметры планировщика *MATLAB DCS* позволяет масштабирование параллельного *MATLAB* кода в кластерах, он включает основной планировщик и также поддерживает *LSF, PBS, TORQUE* и *Windows CCS*

Интерактивный (диалоговый) Параллельный *MATLAB. PCT* обеспечивает способность использовать 4 Рабочих станции в единственной настольной системе, что есть достаточно полезным и удобным для отладки и кодирования разработок. Включение режима выполняется командой - *pmode start local* 4.

Выполнение неинтерактивных заданий. *PCT* может также быть испо-льзован для выполнения неинтерактивных заданий, при этом Задание может быть выполнено Локально, что полезно для разработки прототипа или Удаленно: на кластере в конъюнкция *MATLAB Distributed Computing Server*, при этом существует возможность пропорцио-нального увеличения, чтобы обеспечить значительно большее количество параллельных *Labs*

Функции, которые могут быть использованы для выполнения заданий локально или на кластере.

Базовые команды. *PCT* предлагает следующие 2 функции для вычисления *MATLAB* функции на мульти процессорах: *dfeval*: вычисление функции в клстере; dfevalasync : вычисление функции в кластере асинхронно. Обе функции подобны eval функции, но они влияют на *PCT* с точки зрения вычисления функции на специфицированных компьютерных ресурсах, выход через k-ый *Worker/Lab.*

Данные параллельных заданий. Данные параллельных заданий в общем случае могут быть классифицированы в 2 типа проблем:

возможность: обработать данные в единственном процессоре (с учетом того, что для этого могут потребоваться часы или дни);

В частном случае, тензорное произведение 2-х двумерных квадратных матриц может быть записано: данные, которые должны обрабатываться, слишком большие для единственной системы, Например, в медицинских приложениях формирования изображения, образы могут быть большим как 100000 ×100000.

Параллельная Реализация Данных состоит в их разделении на «куски» данных таким образом, чтобы каждый «кусок» обрабатывался отдельной рабочей станцией.

Используя *PCT*, данные могут быть использованы двумя способами:

1) используется явное прохождение сообщения: *Labs/Workers* могут использовать *MPI*, чтобы распространить данные, все распределения данных должно программироваться пользователем;

2) использование распределенных массивов: *PCT* управляет связью, необходимой для того, чтобы организовать данные через *Labs/Workers*. Пользователь определяет дистрибутивный образец.

Создание распределенных массивов, типы распределений. *MATLAB* поддерживает следующие типы схем распределения:

- 1d – распределение вдоль одного измерения, поддержка для всех массивов. Распределение данных не циклически вдоль одного измерения;

- 2d – распределение вдоль 2-х измерений. Поддержка только для 2D массивов. Распределение матриц вдоль 2-х измерений. Распределение по умолчанию есть 1d –массив, распределенный вдоль колонки.

Связи между рабочими станциями (*Labs*). *PCT* имеет следующие функции для пересылки данных между labs:
labSend – послать данные к lab - labSend(data, destination);
labReceive – получить данные от другой lab - data = labReceive(source); labSendReceive – мгновенно послать и получить данные, эта функция позволяет избежать

ту-пика со связями между labs - data = lab-SendReceive(labTo, labfrom, data).

Параллельные вычисления могут быть описаны последовательностью шагов:

1) от каждого клиента *PC* требуется вычислительная задача, созданная с использованием *PCT*;

2) менеджер работы обеспечит управление задания на конкретных рабочих станциях, где *Matlab MDCE* должен быть установлен;

3) после окончания работ менеджер соберет все ответы от рабочих станций и пошлет их клиенту.

Выводы

Моделирование нечетких переменных тензорами четных рангов позволяет свести все операции нечеткой математики к матричным операциям. Показано эффективность параллельного выполнения операций нечеткой математики в среде *pMatLab*.

Список литературы

1. Воеводин В.В., Воеводин Вл.В. Параллельные вычисления. - СПБ.: БХВ – Петербург, 2002. – 608 с.

2. Минаев Ю.Н., Филимонова О.Ю. Нечеткая математика на основе тензорных моделей неопределенности. Ч. 1 – тензор-переменная в системе нечетких множеств // Электронное моделирование, № 1. – т.30 – 2008. – С. 43 – 59; ч. 2- нечеткая математика в тензорном базисе. – // Электронное моделирование, № 2. – т. 30 - 2008. – С. 4 – 21.

3. Воеводин В.В., Кузнецов Ю.А. Матрицы и вычисления. – М.: Наука, 1984. – 320 с.

4. Беллман Р. Введение в теорию матриц. – М.: Мир, 1972. – 367 с.

5. Мунерман В.И. Параллельная обработка данных методы и средства - Интернет ресурс: http://www.elibrary.ru /item.asp?id=14075.

6. Simon, B. Trace ideals and their applications. - Second Ed., Amer. Math. Soc., 2005. – 265 с.

7. Дилигенский Н.В., Дымова Л.Г., Севастьянов П.В. Нечеткое моделирование и многокритериальная оптимизация производственных систем в условиях неопределенности: технология, экономика, экология М.: Изд-во Машиностроение 1, 2004. – 397 с.

8. Siddharth S. SC08 Engineering Track: Parallel computing using *MATLAB. EMPOWER. PARTNER. LEAD.* – Интернет-ресурс:http://www.SC09.sc-ucation.org/ materials/.../Engineering/Parallel *MATLAB*-SC09.pdf.

9. Kajan S., Sekaj I., Oravec M. The Use of Matlab Parallel Computing Toolbox for Genetic Algorithm-Based MIMO Controller Design, Editors: Fikar, M., Kvasnica, M., In Proceedings of the 17th International Conference on Process Control '09, Štrbské Pleso, Slovakia, P. 277 – 280. – 2009. http://www.kirp.chtf.stuba.sk/pc09.