ДЕКОМПОЗИЦИЯ ДАННЫХ И ВЫЧИСЛЕНИЙ В ЛОКАЛЬНО-РЕКУРСИВНЫХ НЕЛОКАЛЬНО-АСИНХРОННЫХ (LRNLA) АЛГОРИТМАХ

В.Д. Левченко

Введение. Традиционные алгоритмы, занимающие промежуточное положение между численными схемами и их программными реализациями — в настоящее время стали «узким местом», затрудняющим эффективное использование вычислительных ресурсов современных параллельных компьютеров. Ситуация, когда аппаратные решения на десятилетия опережают их алгоритмическую поддержку — выглядит во многом парадоксальной.

Так, развитие иерархии подсистемы хранения данных (от накристальных кэшей до распределённых сетей) стало решением проблемы дисбаланса производительности процессора и оперативной памяти начала 1990х гг. Но в задачах, требующих многократной обработки больших массивов данных, таких как актуальные задачи эволюционного численного моделирования — этот дисбаланс сохранился. Традиционно используемые алгоритмы с пошаговой синхронизацией требуют обработки всего массива данных на каждом шаге численной схемы. Так как все данные задачи заведомо не помещаются в верхних уровнях иерархии памяти, это приводит к нелокальности существенного числа доступов, и как следствие к деградации абсолютной эффективности вычислений (отношение достигнутой производительности к пиковой) с ростом размера обрабатываемых данных вплоть до 10-100 раз (накопленный к настоящему времени уровень дисбаланса производительности CPU и RAM). Наблюдаемое же быстрое наращивание доступных вычислительных ресурсов в гетерогенных суперкомпьютерах с точки зрения традиционных алгоритмов разбиения области выглядит настолько неуниверсальным, что новые параллельные архитектуры рискуют не «дожить» до начала своего массового использования.

В данной работе мы отталкиваемся от тезиса, что причина наблюдаемых проблем состоит в том, что используемые на практике алгоритмы с пошаговой синхронизацией (как последовательные, так и параллельные) существенно устарели и требуют замены. В качестве такой замены предлагается семейство локально-рекурсивных нелокально-асинхронных (LRnLA) алгоритмов, подходящих для широкого класса распространённых численных схем и инструментов программной реализации.

Алгоритмы LRnLA используют общий принцип «разделяй-и-властвуй». Задача, требующая обработки большого массива данных на протяжения тысяч шагов разбивается на ряд подзадач, каждая из которых обрабатывает меньшие массивы данных на меньшем числе шагов. Свойство «нелокальной асинхронности» алгоритмов обеспечивает наличие среди этих подзадач ряда асинхронных, которые могут выполняться параллельно. В более общем виде то же свойство определяет все существующие зависимости по данным между подзадачами. Зависимая подзадача начинает свое выполнение после окончания выполнения тех подзадач, выходные данные которых являются её входными данными. Синхронизацией таких подзадач является «процесс пересылки данных» между ними, при этом требуется в сотни раз меньше синхронизаций по сравнению с методом разбиения области.

Свойство «локальной рекурсивности» LRnLA алгоритмов позволяет производить указанное выше разбиение на подзадачи рекурсивно. При каждом таком разбиении уменьшается обрабатываемый подзадачей массив данных, вследствие чего происходит автоматическая локализация данных в подсистеме кэшей, независимо от деталей её функционирования (свойство cache-oblivious). Для одинаковых численных схем асимптотическая сложность локально-рекурсивных алгоритмов по числу операций совпадает с алгоритмами с пошаговой синхронизацией, но отличается от последних максимальной степенью полинома ($O(D)\sim=D^{1+1/d}$ вместо $O(D)\sim=N_TD$), что обычно нивелирует дисбаланс производительности процессора и темпа доступа к подсистеме памяти.

На практике для актуальных вычислительных систем это означает достижения производительности, близкой к пиковой, что продемонстрировано для широкого класса современных параллельных компьютеров с развитой иерархией памяти, начиная от многоядерных персональных компьютеров и их локальных сетей, кластеров как с «легкими», так и с «тяжелыми» NUMA узлами и заканчивая суперкомпьютерами. С использованием этих алгоритмов решены ряд задач из различных предметных областей, включая кинетическое моделирование плазмы, динамику наноразмерных магнитных доменов, спинтронику, полноволновое моделирование в сейсморазведке, газодинамику. Вместе с тем, вне научной группы автора обоснования алгоритмов не получило особой известности. Настоящая работа призвана частично заполнить этот пробел.

В качестве примера в докладе будут приведены результаты решения двух задач моделирования, аналоги решения которых алгоритмами с пошаговой синхронизации автору неизвестны. Первая — прямое полноволновое моделирование в сейсморазведке на глубину земной коры имеет практические приложения в задачах поиска нефти и газа, вторая — полномасштабное исследование филаментационной неустойчивости в лазер-плазменном взаимодействии — в теории физики плазмы и УТС.

Пространственно-временная локализация с учётом информационных зависимостей. Будем считать, что проблема сформулирована в виде задачи Коши, то есть эволюционной задачи с начальными и граничными условиями на некоторую функцию, определённую в ограниченной области конфигурационного пространства размерности d. В целевых задачах d=3, однако при помощи разрабатываемого подхода возможно решение задач как меньшей, так и большей размерности. Все выкладки проводятся для произвольного значения d, а иллюстрации - для d=1. В конфигурационном пространстве введена декартова система координат. Для решения используется численный метод, имеющий локальный шаблон.

Под «алгоритмами» в этой работе понимается определённый порядок обхода графа зависимостей задачи. Поскольку такой граф принято называть «графом зависимостей алгоритма», тут возможна терминологическая путаница. Чтобы ее избежать для обозначения порядка его обхода используют термин «способ отражения алгоритма на ЭВМ», однако в рамках данной работы такой термин не подходит. Мы отталкиваемся от понятия алгоритма как порядка вычислений, решающих поставленную задачу за конечное время на актуальных вычислительных системах. В этом смысле разным способам обхода одного и того же графа зависимостей из рассматриваемого класса соответствует время решения задачи, отвечающее разным оценкам по размеру данных, от почти линейного до экспоненциального. В рамках конструктивной теории алгоритмов — это не просто разные алгоритмы, а алгоритмы разных классов сложности.

Граф зависимостей задач прямого моделирования эволюции физических сред и систем многих частиц в многомерных по пространству областях, представленный в ярусно-параллельной форме, удобно сопоставить с аналогом физического пространства-времени. При этом номер яруса графа зависимостей сопоставляется с нормированным временем t, а конфигурационную координату x конкретного узла графа зависимостей можно определить по координате данных (например, компонент физического поля), являющихся результатами соответствующей операций. Будем также называть множество всех координат (t,x) пространством операций задачи, а множество всех x — пространством данных.

Чтобы отделить собственные свойства алгоритмов LRnLA от специфики разнообразных решаемых ими задач моделирования и использованных численных схем, для каждой конкретной задачи перенормируем координаты пространств операций и данных таким образом, чтобы за единицу времени информация распространялась не более чем на единицу расстояния. Иными словами, для любых зависимых узлов (t_1x_1) и (t_2x_2) выполнено $|x_2-x_1|<=|t_2-t_1|$. Очевидно, подобное построение актуально лишь для схем с локальным шаблоном, что и является основным условием применимости алгоритмов LRnLA. Также введем понятия плотности операций (данных) как число элементарных операций (данных) численной схемы, соответствующих единичной ячейке пространства операций (данных).

Наша задача будет состоять в том, чтобы, учитывая обычные требования численных методов, провести дискретизацию пространства операций на подобласти, упорядочив зависимости между ними. В качестве максимального времени отслеживания зависимостей (высотой конуса) зададимся некоторым промежутком времени T/2 (рис.1а), с одной стороны, достаточно большим по сравнению с обычной дискретностью времени моделей, а с другой - достаточно маленьким по сравнению с полным временем моделирования. Промежуток времени T будем называть шагом синхронизации, и на его протяжении потребуем отсутствия операций ввода-вывода. Основаниями конуса в моментах синхронизации будет d-мерный шар с диаметром T. Далее нам потребуются геометрические фигуры, покрывающие область моделирования. Так как в данной работе мы будем исходить из кубических сеток, то впишем полученный шар в d-мерный куб с рёбрами, параллельными осям координат декартова пространства.

Сдвигая (транслируя) полученный куб вдоль координатных осей, покроем всю область моделирования кубической сеткой G, для узлов и ячеек которой будем использовать обозначения $X_{d,i}$ и $O_{d,i}$, i — индекс ячейки. Для удобства введем также сетку G', сдвинутую относительно G на T/2 по всем осям пространства-времени. При этом узлы основной сетки совпадут с центрами ячеек дополнительной и наоборот. LRnLA ячейкой назовём подобласть (d+1)-мерного пространства, соединяющую ячейки введённых сеток (рис 1а).

Для узлов и ячеек, выходящих на границу области моделирования потребуется ввести другие обозначения, этот вопрос требует отдельного рассмотрения

Конусоиды зависимости и конусоиды влияния. Конусоидом зависимости узла сетки G на шаге синхронизации k назовем (d+1)-мерную пирамиду в пространстве-времени с вершиной в этом узле и основанием, совпадающим с ячейкой сетки G' в момент времени, отстоящий на -T/2. Аналогично, конусоидом влияния того же узла назовем «перевёрнутую» (симметричную относительно вершины) пирамиду, то есть её основание находится в момент времени, отстоящий на +T/2 (рис.1б). Введем специальное обозначение для конусоидов зависимости и влияния, использующие введённые ранее обозначения узлов, для которого строится конусоид (рис 1б).

Основным полезным свойством конусоидов (свойством локальности), отражённом и в их названии, является следующее:

- любая точка, принадлежащая конусоиду зависимости, не может зависить от точек, находящихся вне его, начиная с момента синхронизации;
- любая точка, принадлежащая конусоиду влияния, не может влиять на любую точку, находящихся вне его вплоть до момента синхронизации.

Конусоиды зависимости и влияния узлов, имеющие координату i и моменты синхронизации соответственно k и k+1, а также k+1/2 - принадлежат соответствующей LRnLA ячейке (рис.1б), но, при d>1 не заполняют её полностью. То есть для получения разбиения пространственно-временной области необходимо проделать дополнительные построения.

Отметим, что несмотря на то, что данные фигуры геометрически являются пирамидами, для них специально выбрано название «конусоид», чтобы подчеркнуть связь с конусом Минковского, нижняя и верхняя половинки которого имеют аналогичные свойства. В дальнейшем понятие конусоидов будет распространено на фигуры более сложной формы, конусоиды можно строить для треугольных сеток, решетчатых графов и т.д. - первичным во всех этих случаях являются информационные связи. Введение конусоидов посредством тех или иных геометрических фигур в данной работе в основном используется как наглядная иллюстрация их свойств, в то время как под конусоидами в дальнейшем будут пониматься алгоритмы (правила обхода графа зависимостей задачи).

В частности, для явных сеточных схем в многомерной области максимальная скорость распространения возмущения отличается от своего физического аналога. Более того, в задачах эллиптического и параболического типа нет ограничения на скорость распространения возмущений и, соответственно, аналогов конуса Минковского. Однако, информационные зависимости для некоторых итерационных методов решения этих задач совпадают с рассматриваемыми ниже, и тогда для них также можно использовать LRnLA алгоритмы.

Конусоиды зависимости-влияния. Введем обозначения для элементов поверхности ячейки сеток G и G'. Эта поверхность состоит из набора кубов меньшей размерности, промежуточной между d-кубом (внутренним объёмом ячейки сетки) и 0-кубом (точкой, узлом сетки). Как и раньше, пару элементов сеток G и G', имеющих один и тот же центр, будем называть дополнительными (друг к другу). Отметим, что для каждого элемента найдется единственный дополнительный элемент, причём их декартово произведение даст d-мерный куб.

Для каждого из базовых элементов сеток можно определить конусоид зависимости и конусоид влияния (рис.1в) как объединение соответствующих конусоидов составляющих эти элементы точек.

Теперь для пары элементов сеток, отстоящих друг от друга по времени, введем основное в данной работе понятие конусоида зависимости-влияния (или просто конусоида) как пересечение конусоида влияния одного элемента сетки (нижнего основания) с конусоидом зависимости другого элемента (верхнего основания). Элементарным конусоидом (ConeTur) назовем конусоид, нижнее и верхнее основания которого являются дополнительными элементами сеток на соседних моментах синхронизации (рис.1в).

ConeTur, в обозначении которых присутствует одинаковый набор символов (равное число X и O) имеют одну и ту же форму как (d+1)-мерные фигуры в пространстве-времени. В этом смысле будем говорить о форме конусоида, и, при фиксированной размерности d характеризовать эту форму одним целым числом S, равным по модулю числу символов O в полной символьной кодировке ConeTur. При этом, если синхронизация по времени находится в прошлом, то это число отрицательное, а если в будущем - положительное. Кроме того, заметим, что пара конусоидов форм -0 и +0 имеют основание d-куб в один и тот же момент времени, а их объединение является в свою очередь конусоидом зависимости-влияния, построенном на узлах сетки, относящихся к соседним целым моментам времени. Конусоид такой формы и будем обозначать числом 0. Итого, S пробегает значения -d,...,0,...,d - всего (2d+1) значений (или форм ConeTur). Несложно также подсчитать число уникальных кодировок для ConeTur каждой формы. Учитывая, что в символьной кодировке всего d позиций, в которых расположено s символов O и (d-s) символов X, получаем число сочетаний C_d^s. Заметим, что ConeTur одной формы, но с разной символьной кодировкой можно получить перестановкой осей координат, в этом смысле будем говорить о них как о ConeTur разных видов одной формы. Так как размещение пары символов O и X в d позициях возможно 2^d способами, а синхронизация каждого вида ConeTur может производиться двумя способами (в прошлом или будущем), за исключением формы S=0, то полное число разных видов разных форм элементарных ConeTur равно 2^{d+1}-1.

Введение элементарных конусоидов позволяет от проблемы выяснения отношений зависимости и влияния между отдельными точками пространства-времени перейти к проблеме определения этих соотношений между ними.

Свойства ConeTur По построению, ConeTur являются выпуклыми фигурами размерности d+1 и ограничены гиперплоскостями размерности d. Эти гиперплоскости относятся к двум типам: моменты синхронизации и проекции информационных зависимостей по каждой координате. Разбиение области пространства-времени указанными выше гиперплоскостями дает систему ConeTur или, другими словами, введённое выше семейство ConeTur определяет простое разбиение области пространства-времени.

По построению ConeTur, между моментами синхронизации, он зависит от своего дополнения до производящего конусоида зависимости, и влияет на своё дополнение до производящего конусоида влияния. Непосредственно зависимыми будем называть конусоиды, имеющие общие границы размерности d, и обозначать стрелкой (рис.2б).

Для идентификации ConeTur будем использовать следующие признаки: целое число формы, символьный код вида, и пару координат. На уровне формы зависимости между ConeTur определяются рядом целых чисел: -d,...,0,..,d,-d,... Коды формы зависимых ConeTur отличаются одним символом (в зависимости от знака S, X меняется на O или O на X); очевидно, что число возможных изменений, равно числу таких символов в коде, а именно |S| или d-|S|. Каждому изменению кода соответствует пара ConeTur с координатами, отличающимися на 1. Таким образом, общее количество зависимых или влияющих ConeTur равно 2|S| или 2(d-|S|).

Свойство локальности вынесено в заглавие семейства алгоритмов LRnLA вследствие практической важности соответствующего свойства алгоритмов, выражающегося в отношении количества вычислений и обрабатываемых данных. Определим параметр локальности для конусоидов как отношение его объёма к площади его проекции, нормированное на отношение плотности операций к плотности данных. Для ConeTur формы 0 получим значение T/(d+1), для остальных форм - вдвое меньше.

Свойство асинхронности наряду с предыдущим, является одним из двух наиболее важных для практического применения свойств и выражает число асинхронных подзадач (а также пределы ускорения на идеальном параллельном компьютере).

Посчитаем количество независимых (то есть асинхронных) ConeTur в области моделирования. В каждой LRnLA ячейке независимыми являются ConeTur одной формы разных видов. Как уже было замечено ранее, для формы S их $C_d^{|S|}$ штук. Кроме того, ConeTur, относящиеся к LRnLA ячейкам с разными индексами i и одинаковыми k также асинхронны. То есть, если в области N_c ячеек, то число асинхронных ConeTur равно $N_c\, C_d^{|S|}$.

Локально-рекурсивная декомпозиция Для перехода к более мелкой сетке будем использовать (d+1)-бинарное разбиение области пространства-времени. Для этого разделим интервал синхронизации T пополам, также как и каждый интервал O по каждой оси координат. В результате ячейки сеток G и G' разделятся на 2^d ячеек более мелких сеток каждая. На новых сетках также построим систему ConeTur (рис.2в). Уровень крупности сеток будем обозначать целым числом (рангом) r. При этом ConeTur более крупного ранга разбивается на несколько ConeTur более мелкого ранга, правила которого можно также определить, используя введённую ранее кодировку форм и видов. ConeTur формы S ранга r разбивается на набор ConeTur (d+1) форм (в случае S=0 (2d+1) форм), а зависимости между ConeTur ранга (r-1) определяются введённой ранее формулой со следующими дополнениями: коды форм, на которые разбивается ConeTur, пробегают ряд целых чисел в порядке роста между (S-d sign(S)) и S включительно (в случае S=0 получается 0,...,d,-d,...,0), а количество ConeTur каждой из этих форм S равно $2^{|S|}$. Коды форм, а также координаты ConeTur определяются по уже введённым правилам.

Описанное разбиение ConeTur будем проводить рекурсивно вплоть до ранга r=0. На этом уровне происходит замыкание алгоритма на выбранную численную схему, которое состоит в привязке определённого действия численной схемы, связанное с обновлением данных — к конкретному ConeTur, например, с помощью привязки к координатам пространства-времени и отслеживания зависимостей по шаблону численной схемы. Для реализации схем с высокой локальностью, таких как конечно-разностные методы или методы крупных частиц второго порядка точности, в том числе и на сдвинутых сетках, достаточно поставить в соответствие ячейку численной схемы и ячейку LRnLA. Напомним, что каждой ячейке LRnLA соответствует 2^{d+1}-1 разных ConeTur, чего обычно заведомо достаточно. Более того, часть ConeTur ранга 0 при этом могут оставаться пустыми (не производящими никаких действий в рамках выбранной численной схемы). Для менее локальных схем, таких как схемы повышенного порядка точности, с коррекцией потока, многошаговые, Рунге-Кутты и т.д., требуется объединить несколько ячеек численной схемы в одну LRnLA ячейку.

Классификация LRnLA алгоритмов Множество графов, получающихся рекурсивным разбиением исходного графа зависимостей задачи при помощи ConeTur, удобно трактовать как обобщённый граф зависимостей (ОГЗ) LRnLA. Тогда любой LRnLA алгоритм можно определить как допустимый порядок обхода узлов ОГЗ. Число таких возможных обходов велико, но их можно упорядочить, введя классификацию, построенную на следующих признаках. *Класс:* алгоритмы с принципиально разным набором свойств, объединены общей идеей построения; *Размерность*: размерность декомпозиции пространства данных d, равна или меньше размерности конфигурационного пространства; *Форма*: правила композиции/декомпозиции обобщенного графа зависимостей LRnLA, формула кодировки; *Вид*: в рамках одной формы отличаются перестановками индексов координат, в формуле кодировки имеют одинаковый набор символов, но в разных позициях; *Ранг*: номер уровня рекурсии (целое число r), определяет размер конусоида, 2^r коэффициент подобия; *Координаты*: координата по времени — целое число k, номер шага текущего ранга, координата (или координаты) по пространству, достаточно одного целого числа — номера ячейки сетки текущего ранга в конфигурационном пространстве.

Алгоритмы класса ConeFold Из-за большого количества форм и видов ConeTur, их программная реализация связана с определёнными сложностями, особенно для императивных языков программирования. Однако, взяв по одному ConeTur каждого вида каждой формы, можно сложить конусоид, одной формы и вида

которого достаточно для заполнения всего пространства операций. Такой класс конусоидов назовем ConeFold (рис.3а).

ConeFold нулевого ранга обычно содержит вычисления для одной ячейки использованной численной схемы. Это с одной стороны, упрощает до тривиальности разработку реальных численных схем с использованием ConeFold-а, а с другой - уменьшает накладные расходы при реализации LR обхода. Дополнительно, коэффициент локальности совпадает с наилучшим из элементарных конусоидов. Кроме того, очень прост алгоритм LR декомпозиции, который для ConeFold является d+1-бинарным, иными словами, каждая из сторон ConeFold делится пополам, а проведённые d+1 сечений дают 2^{d+1} ConeFold меньшего на 1 ранга.

Определим число асинхронных ConeFold в области моделирования. Основным отличием от ConeTur является то, что синхронизированные по времени ConeFold зависимы по цепочке, то есть число ярусов равно числу ConeFold в наборе. Независимыми же является набор ConeFold, синхронизированный «ступеньками лесенки» с «высотой» каждой ступеньки по времени, равной 1 и наклоном к оси времени 1:2 вдоль координатных осей. Число ярусов при этом равно d+1. Такой набор может давать 100% эффективность при распараллеливании, как и в случае синхронизированного по времени набора ConeTur. В отличие от последнего, число процессоров, на котором достигается максимальное ускорение (как и само ускорение) в d+1 раз ниже. Однако обычно этот параметр имеет большой запас по отношению к имеющейся аппаратной параллельности.

На основе ConeFold построены алгоритмы с различными свойствами: максимальной локальности (ConeTorre), максимальной асинхронности (ChessFold), и их комбинация (TorreFold) (рис.3б-г).

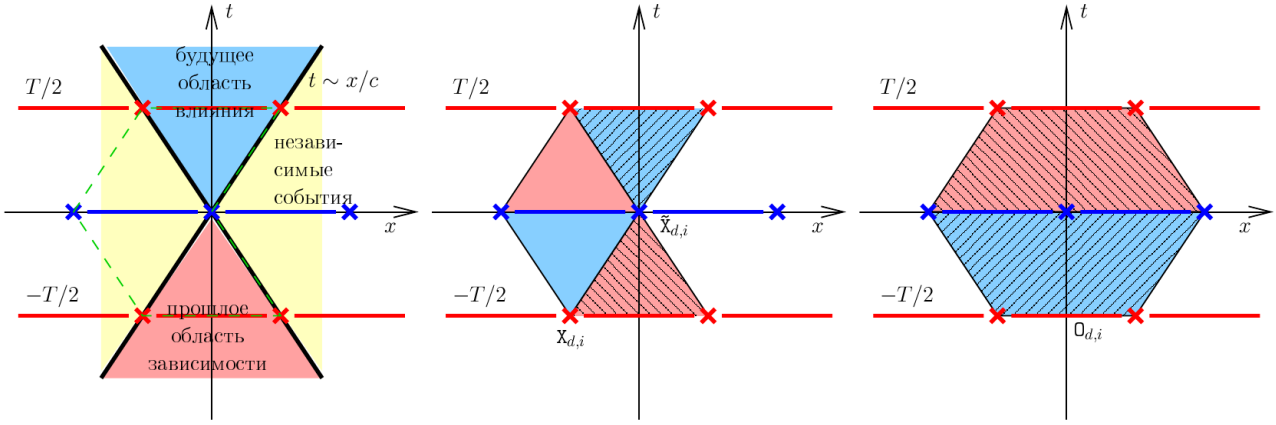

Рис.1. а) Основная (красная) и дополнительная (синяя) сетки; LRnLA ячейка выделена зелёным. б) Конусоиды зависимости (розовый) и влияния (светло-синий) узлов основной и дополнительной (штриховка) сеток. в) Элементарные конусоиды (голубой) и (малиновый)

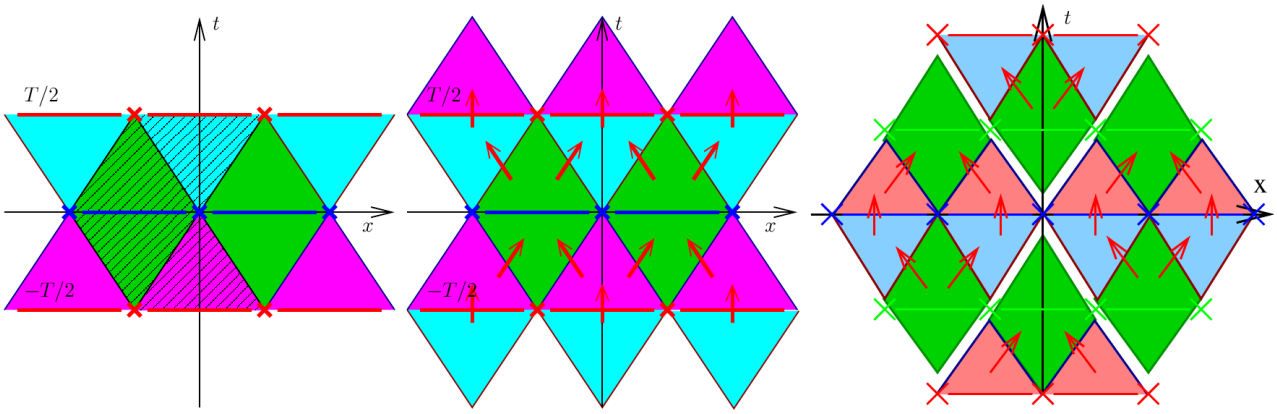

Рис.2. а) Разбиение LRnLA ячейки; разные формы ConeTur отмечены разными цветами. б) Зависимости между элементарными конусоидами, асинхронные конусоиды имеют один цвет. в) Декомпозиция ConeTur

Рис.3. а) Разбиение ConeFold. б-г) Графическое изображение алгоритмов ConeTorre. в)

Заключение В работе введена система высокоуровневых абстракций, позволяющая разрабатывать новые алгоритмы семейства LRnLA решения задач моделирования, эффективные для широкого класса современных вычислительных систем, обладающих развитой иерархией подсистемы памяти и параллелизма. Введена классификация алгоритмов и основные параметры локальности и асинхронности, количественно выражающие их свойства. Приложения, созданные с использованием введённого формализма, а также модель вычислений будут представлены на конференции в других докладах.

Работа поддержана грантом РФФИ 12-01-00708.

ЛИТЕРАТУРА:

1. В.Д. Левченко. "Асинхронные параллельные алгоритмы как способ достижения эффективности вычислений" // Информационные технологии и вычислительные системы, 2005 (1)