УДК 681.513

И.А. Осотов, О.Ф. Михаль
ХНУРЕ, г. Харьков, Украина, fuzzy16@pisem.net

ОЦЕНКА ЭФФЕКТИВНОСТИ ПРОВЕДЕНИЯ ЛОКАЛЬНО-ПАРАЛЛЕЛЬНЫХ ВЫЧИСЛЕНИЙ НА МНОГОЯДЕРНЫХ ПРОЦЕССОРАХ

На основе анализа тенденций развития процессорной техники обоснована перспективность расширенного использования локально-параллельных методов обработки. В машинном эксперименте реализован выигрыш в эффективности ~15% для локально-параллельного алгоритма с потоком обработки данных внутри распараллеленного участка.

ЛОКАЛЬНО-ПАРАЛЛЕЛЬНАЯ ОБРАБОТКА ИНФОРМАЦИИ, МНОГОЯДЕРНЫЕ ПРОЦЕССОРЫ

Введение

Объекты живой природы, в том числе *живые системы* (ЖС), наделённые интеллектом (в различной степени), по комплексу характеристик надёжности, устойчивости, способности к самообучению и др. существенно превосходят образцы *технических систем* (ТС), разработанных человеком. При этом сами ЖС собраны из неустойчивых и ненадёжных компонентов, то есть высокие *эксплуатационные характеристики* (ЭХ), включая интеллектуальный аспект, достигнуты в них за счёт системных и структурных решений. В связи с этим актуален круг проблем по системному исследованию интеллектуальных ЖС для использования соответствующих структурных решений в ТС. В [1, 2] рассмотрены иерархические параллельно-организованые ЖС с ретроспективой на тенденции развития *вычислительных устройств* (ВУ). Показано, что перспективно использовать ВУ общего назначения с программной реализацией на них *локально-параллельных* (ЛП) методов обработки информации. Новый аспект применения принципа ЛП связан с многоядерностью процессоров ВУ. На настоящий момент ЛП обработка на мнгоядерных процессорах недостаточно изучена. В частности для известных [1, 2] ЛП алгоритмов не разработаны версии их выполнения на многоядерных процессорах; ввиду чего уже разработанные [3, 4] методы исследования для многоядерных процессоров не апробированы на известных ЛП алгоритмах. Между тем, ожидаемый прирост эффективности составляет десятки процентов, чем определяется целесообразность проведения разработок в данном направлении.

Целью настоящей работы является оценка эффективности проведения ЛП вычислений для конкретных типов алгоритмов на многоядерных процессорах.

1. Иерархичность и параллеленость

В плане обеспечения надёжности и устойчивости функционирования ЖС (объекты живой природы) являются неоспоримыми (не теряющими актуальность) прототипами для ТС. Свойства ЖС по самовосстановлению, авторепродуцированию, обучаемости, приспособляемости к агрессивным внешним условиям и др. по настоящее время в технических устройствах не воспроизведены. Базовые биологические (высокомолекулярные) материалы, из которых собраны ЖС, имеют малую устойчивость и критичны к условиям внешней среды. Тем не менее, вследствие иерархической структурной организации ЖС демонстрируют долговременное устойчивое функционирование и приспособляемость – сохранение *эксплуатационных характеристик* (ЭХ) в широких диапазонах по ряду параметров (P_1, P_2,..., P_j,...), задаваемых внешней средой. В этом они значительно превосходят менее сложно организованные, и потому казалось бы более надёжные, ТС. Важнейшими особенностями обеспечения высоких ЭХ в ЖС, являются многоуровневость (иерархичность) и параллельная организация в каждом из уровней (внутриуровневый локальный параллелизм).

Упрощённо иерархичность и распараллеленность ЖС выглядят следующим образом. Имеются объекты ЖС и внешняя среда, в которой находятся ЖС. Каждый из объектов ЖС является многоуровневым по сложности. На i-м уровне сложности ЖС состоит из функционально однотипных объектов, каждый из которых состоит из объектов (i-1)-го уровня. Понятие «внешняя среда» – относительное: объект (i+1)-го уровня является внешней средой для объектов (i-1)-го уровня. Поэтому группы объектов (i-1)-го уровня объединяются в объекты i-го уровня сложности с целью противостоять воздействиям внешней среды. Для этого на i-м уровне однотипные элементы (i-1)-го уровня взаимоорганизуются так, что в целом на i-м уровне обеспечиваются обратные связи, поддерживающие объект i-го уровня в рамках параметров ($p_{min,j,i}$; $p_{max,j,i}$), где в индексах: j – номер параметра внешней среды P_j; i – номер уровня иерархической организации. При этом для ЖС вцелом по j-му параметру внешней среды P_j обеспечивается вложенность интервалов значений параметров:

$$p_{min,j,1} \geq p_{min,j,2} \geq p_{min,j,3} \geq \ldots \geq p_{min,j,(i-1)} \geq p_{min,j,i} \geq \ldots;$$
$$p_{max,j,1} \leq p_{max,j,2} \leq p_{max,j,3} \leq \ldots \leq p_{max,j,(i-1)} \leq p_{max,j,i} \leq \ldots . \quad (1)$$

В результате оказывается, что имеется последовательная вложенность «сред обитания» объектов

«друг в друге», подобная игрушке «Матрёшка». При этом объект иерархии ЖС, находящийся на верхнем уровне, эффективно функционирует в «самой внешней» «истинно окружающей» среде, разрушительной для исходных материалов (базовый уровень), из которых собрана ЖС.

Человек является частным случаем (одним из видов) ЖС, поэтому приведённое описание (системный генезис) плюс продолжение на социальный аспект развития относятся также и к нему. Деятельность человека в системном плане направлена на расширение своих возможностей и совершенствование выполняемых функций. Поэтому разрабатываемое человеком искусственное (рукотворное) окружение (от примитивных орудий труда до ВУ и средств телекоммуникации) является его антропоморфным продолжением: усилением возможностей, улучшением системных ЭХ. Исходя из этого, целесообразны разработки с соблюдением принципов иерархичности и параллелености в каждом из уровней. Этим, вчастности, обеспечиваются надёжность, взаимозаменяемость, возможность реконфигурирования при частичных отказах, а также дополнительная функциональность за счёт параллельного (фонового) использования резервных возможностей.

Такова общая картина, часто остающаяся вне поля зрения за техническими подробностями. В аспекте разработки ВУ, как интеллектуального элемента человеческого окружения, рассмотрим некоторый частный случай: «внедрения» (проникновение) нового программного распараллеливания внутрь существующего аппаратного уровня. Особенностью является то, что ВУ многоядерное и рассматривается внутриуровневая *программная* распараллеленность на уровне, который сам уже является *аппаратно* распараллеленным.

2. Локальная параллельность

Принцип ЛП обработки информации рассмотрен в [1, 2]. Эффективность ЛП обработки иллюстрируется следующим алгоритмическим примером. Пусть имеется n выражений $c_i = a_i + b_i$; $i = 1, 2,..., n$. Значения a_i и b_i заданы. Требуется найти значения c_i. На одноядерном ВУ данная задача разрешима за $4n$ шагов: по каждому из выражений загрузить a_i, загрузить b_i, произвести суммирование, выгрузить результат c_i. Согласно ЛП принципу значения a_i и b_i конкатенируются:

$$A = a_1 \oplus 0 \oplus \therefore a_2 \oplus 0 \oplus ... \oplus 0 \oplus a_i \oplus 0 \oplus ... \oplus 0 \oplus a_n;$$

$$B = b_1 \oplus 0 \oplus b_2 \oplus 0 \oplus ... \oplus 0 \oplus b_i \oplus 0 \oplus ... \oplus 0 \oplus b_n; \quad (2)$$

а совокупность всех n результатов c_i получается как конкатенация

$$C = c_1 \oplus 0 \oplus c_2 \oplus 0 \oplus ... \oplus 0 \oplus c_i \oplus 0 \oplus ... \oplus 0 \oplus c_n \quad (3)$$

в ходе выполнения 4-шаговой операции

$$C = A + B. \quad (4)$$

Разделительные нули в (2) и (3) вставлены между конкатенантами для реализации переноса разряда при суммировании. Числа a_i и b_i должны быть положительными и иметь ограниченную разрядность, так чтобы конкатенации (2) и (3) не превышали по длине в битах разрядность процессора.

Разумеется, n-кратный выигрыш достигнут без учёта затрат времени на операции конкатенации-деконкатенации, окаймляющие вычислительный блок. Если речь идёт о короткой вычислительной процедуре типа (4), то ЛП вариант реализации скорее всего будет проигрышным по времени ввиду наличия вспомогательных операций формирования (2) и извлечения результатов c_i из (3). Но в реальных расчётных задачах, когда вычислительный блок существенно сложнее (4) и включает более значительный набор разнообразных операций, выполняемых над наборами данных без промежуточных конкатенации-деконкатенации, выигрыш в производительности может быть значительным. Выигрыш растёт с ростом разрядности процессора, а также со снижением точности представления данных («загрублением» системы) при сокращении размеров конкатенируемых сегментов. Оба указанных направления имеют конкретные технические приложения.

Рост разрядности процессоров является долговременной устойчивой тенденцией развития процессорной техники. Разрядность регистра процессора – это объём непосредственно адресуемой оперативной памяти. В конечном счёте, объёмом памяти определяется предельно допустимая размерность задач, которые могут быть разрешены на данном ВУ за приемлемое время. Развитие техники ВУ стимулирует появление новых вычислительных задач, которые в свою очередь стимулируют рост требований к техническим характеристикам вновь разрабатываемых ВУ. Данная петля обратной связи исправно функционирует уже полвека и нет принципиальных причин отхода от данной парадигмы.

Снижение точности представления данных до приемлемых пределов допустимо и целесообразно, в частности, в системах, базирующихся на использовании экспертных знаний. Носителями исходных (первоначальных) экспертных знаний являются люди. В связи с этим должен приниматься во внимание известный из психологии факт [5]: человек может устойчиво различать лишь малое число объектов: 5 ± 2 шт. При работе с большим числом объектов различение идёт с разделением на подуровни: предъявляемые объекты сначала группируются в блоки по $3 - 7$ шт., затем последовательно осуществляется обработка блоков. Таким образом, для кодирования одного из реально различаемых экспертом состояний требуется до 3 бит информации. Использование более высокой разрядности может предполагать округление, сглаживание. То есть, возможно искажение исходных экспертных знаний.

Разумеется, различение 5 ± 2 объектов есть антропоморфная ЭХ. В дальнейшем в ходе эволюции ТС могут быть востребованы системы с

ЭХ, превосходящими антропоморфные. Но по-видимому это будет уже *чисто техническая* эволюция (эволюция ТС, реализуемая не людьми, а самими ТС), которая, возможно, уже не будет относиться к человеческой (*чисто человеческой*) истории. В тот отдалённый период человек скорее всего не будет являться единственным носителем экспертных знаний, и сама проблематика обработки информации приобретёт другие очертания. Текущий же этап развития ТС был и остаётся антропоморфным, и для него остаются справедливыми две сформулированные мотивации по расширенному использованию принципа ЛП:

– рост разрядности процессоров;

– ограниченность разрядности представления экспертных знаний.

Далее рассмотрим ЛП в плане практической реализации на многоядерных процессорах.

3. Многоядерные процессоры

В настоящее время разрядность процессоров общего назначения возросла до 64 бит. Имеются также специализированные графические процессоры, применяемые, в частности, в игровых приставках, имеющие разрядность 128 и 256 бит. При этом тактовая частота ядра процессора достигла единиц ГГц. Дальнейшая конъюнктура рынка процессоров, по-видимому, складывается так, что конкурентоспособный рост производительности ВУ не обеспечивается ростом разрядности регистров процессора. Для основной массы задач общего назначения разрядность 64 бита является достаточной для представления данных, а аппаратное ЛП-представление данных, по видимому, не обеспечивается существующими инженерными (схемными и конфигурационными) решениями в гигагерцевом диапазоне. В техническом отношении подобная задача, разумеется, является нетривиальной: речь идёт о микроминиатюризации в области СВЧ, где помимо чисто схемных решений существенны также и конфигурационные факторы. Вместе с тем процессорная техника должна развиваться, и характеристики должны наращиваться, поскольку развёрнуто и профинансировано несколько конкурирующих организационных структур, занимающихся разработкой и производством процессоров. В сложившейся ситуации, по-видимому, оптимальным по сложной группе критериев «капиталовложения – сроки разработки – выигрыш в производительности» является курс на создание многоядерных однокристальных процессоров. Как известно [6], данное направление было впервые (прорывно, но преждевременно) реализовано специалистами фирмы Digital Equipment Corporation ещё в 1992 г. в виде 64-разрядного RISC процессора DEC-Alpha.

Перспективность однокристальных многоядерных процессоров в техническом отношении состоит в том, что каждое из ядер может быть высокочастотным, а взаимодействие между ядрами реализуется на более низких частотах через кеш-память 2-го, 3-го и т.д. уровней. Таким образом, иерархические уровни образуются здесь по вложенности, объёмам памяти и убыванию рабочих частот. Вложенность уровней кеш-памяти соответствует (аналогична) изложенной выше (п. 1) концепции об иерархичности ЖС, а последовательность частот, убывающая по уровням кеш-памяти, – вложенности диапазонов значений параметров (1).

Могут также реализовываться различные иерархии дисциплин по автономному или совместному использованию различных уровней кеш-памяти. Представляется достаточно вероятным, что прогресс по числу ядер будет сопровождаться прогрессом по организации многоуровневой кэш-памяти, направленным на повышение рабочих частот и усиление обобществления памяти для эффективного взаимодействия между отдельными ядрами.

В целом переход от одноядерной модели процессора к многоядерной определился, повидимому, технологическими ограничениями по освоению СВЧ-диапазона на микроэлектронном уровне, т.е. отсутствием инженерного опыта и разработанных технических решений. Поэтому в конечном счёте путь развития в дальнейшем будет определяться накоплением инженерного опыта при разработке процессорных систем по мере освоения СВЧ-диапазона.

В рамках описанного (возможно, ограниченного) понимания перспектив развития процессорной техники сегодняшние многоядерные однокристальные процессоры предстаются как промежуточный этап перед полностью СВЧ-частотными процессорами с большой разрядностью регистров и реконфигурируемостью по числу ядер (вычислительных узлов). В самом деле: n-ядерный однокристальный процессор с разрядностью каждого ядра m может быть эквивалентен nm-разрядному одноядерному процессору, если обеспечена связь (передача данных) на тактовой частоте ядер между старшим разрядом i-го и младшим разрядом (i+1)-го ядер. В этом случае регистры отдельных ядер процессора конкатенированы в гиперрегистры виртуального единого гиперядра.

В связи со сказанным перспективна реализация ЛП алгоритмов на многоядерных структурах. Технически и программно – ЛП и многоядерность не противоречат друг другу, а в идейном отношении они направлены на реализацию единой цели: внутриуровневой распараллеленности системы.

Рассмотрим далее комбинированное ЛП многоядерное алгоритмическое решение на примере ЛП алгоритма операции нахождения нечёткого теоретико-множественного пересечения двух наборов значений функций принадлежности [1, 2].

4. Алгоритмы

В табл. 1 приведено пошаговое описание ЛП алгоритма нечёткой теоретико-множественной

операции пересечения. Операнды A и B являются конкатенациями масштабированных значений функции принадлежности [2, 3]. Количество и длина сегментов (на примере 10-сегментного 3-битового варианта, удобного для реализации на 32-разрядных ВУ) определяются константами

$E_1=119304647_{10}=00011100011100011100011100011111_2$,

$E_2=954437176_{10}=11100011100011100011100011100000_2$,

с использованием которых выделяются множества нечётных и чётных (соответственно) сегментов. Числа 10 и 2 в индексах при значениях E_1 и E_2 обозначают системы счисления. Константы L_1 и L_2 предназначены для помещения «сигнальных» единиц в младшие разряды сегментов. В 10-сегментном 3-битовом варианте – $L_1=1090785344_{10}$; $L_2=653667127_{10}$.

Таблица 1

Пошаговое описание последовательного варианта локально-параллельной операции нечёткого теоретико-множественного пересечения

Шаг	Метод реализации $min(A, B)$
1	Прореживание: $A_1=A\ and\ E_1$; $A_2=A\ and E_2$; $B_1=B\ and\ E_1$; $B_2=B\ and\ E_2$.
2	Установка меток и арифметическое вычитание: $C_1=(A_1\ or\ L_2)-B_1$; $C_2=(A_2\ or\ L_1)-B_2$.
3	Сопряженное прореживание: $C_1=C_1\ and\ L_2$; $C_2=C_2\ and\ L_1$; или $C_1=C_1\ and\ E_2$; $C_2=C_2\ and\ E_1$.
4	Склеивание: $C=C_1$ "or" C_2.
5	Формирование маски: $M=C-(C>>n)$.
6	Формирование инверсной маски: $\bar{M}=M\ xor\ E_0$.
7	Результат:
	$(A\ and\ \bar{M})\ or\ (B\ and\ M)$;

Смысл остальных обозначений в табл. 1 следующий: $E_0=E_1+E_2$; $L_0=L_1+L_2$; *and*, *or* и *xor* – битовые логические операции; $(C>>n)$ – битовый сдвиг регистра процессора, содержащего C, на n бит в сторону младших разрядов с потерей содержимого n младших разрядов; при 3-битовых сегментах $n=3$.

После склеивания (Шаг 4) формируются прямая и инверсная маски, с использованием которых собирается результат (Шаг 7).

Легко видеть, что Шаги 1-3 содержат по две группы действий: производимые с чётными и нечётными сегментами. Эти группы являются алгоритмически распараллеливаемыми. Затем в Шагах 4-7 результаты объединяются и вычисления идут последовательно.

Двухъядерный вариант алгоритма представлен в табл. 2. Отличие от варианта табл. 1 – разнесение параллельных операций в параллельно выполняемые Шаги. Таким образом, на одноядерных процессорах весь алгоритм может выполняться последовательно (таблица 1), а на многоядерных – Шаги 1.1, 2.1, 3.1 выполняются параллельно с Шагами 1.2, 2.2, 3.2, затем остальная часть – последовательно (табл. 2). В результате, на многоядерных процессорах потенциально достижим дополнительный прирост эффективности – сокращение времени обработки.

Таблица 2

Пошаговое описание варианта локально-параллельной операции нечёткого теоретико-множественного пересечения с распараллеливанием

Шаг	Метод реализации $min(A, B)$
1.1	Прореживание: $A_1=A\ and\ E_1$; $B_1=B\ and\ E_1$.
1.2	Прореживание: $A_2=A\ and\ E_2$; $B_2=B\ and\ E_2$.
2.1	Установка меток и арифметическое вычитание: $C_1=(A_1\ or\ L_2)-B_1$.
2.2	Установка меток и арифметическое вычитание: $C_2=(A_2\ or\ L_1)-B_2$.
3.1	Сопряженное прореживание: $C_1=C_1\ and\ L_2$; или $C_1=C_1\ and\ E_2$.
3.2	Сопряженное прореживание: $C_2=C_2\ and\ L_1$; или $C_2=C_2\ and\ E_1$.
4	Склеивание: $C=C_1$ "or" C_2.
5	Формирование маски: $M=C-(C>>n)$.
6	Инверсная маска: $\bar{M}=M\ xor\ E_0$.
7	Результат:
	$(A\ and\ \bar{M})\ or\ (B\ and\ M)$;

Разработанная ранее [3, 4] методика машинного эксперимента для анализа эффективности была адаптирована с учётом структурной схемы ЛП алгоритма (табл. 1, 2). Согласно плану эксперимента замеры затрат времени на выполнение проводились по фиксациям моментов времени для одно- и двухядерного вариантов реализации алгоритма при одинаковом числе итераций во внутреннем цикле.

Коэффициент эффективности определяется как отношение времён выполнения одноядерного варианта (аппаратно-последовательного) к двухъядерному (с внутрипроцессорным аппаратным распараллеливанием). То есть, если коэффициент превышает единицу – имеется выигрыш по эффективности.

Программная реализация механизма внутрипроцессорного распараллеливания рассмотрена в [3, 4]. Применены стандартные языковые конструкции инструментария WinAPI: функции `SetThreadAffinity-Mask` и `SetThreadIdeal-Processor`.

На рис. 1 представлена блок-схема программы для исследования роста эффективности распараллеленного варианта алгоритма (табл. 2) по сравнению с последовательным (табл. 1). В левом столбце представлены преимущественно обслуживающие процедуры: ввод исходных данных, генерация массивов значений, организация внешнего цикла накопления результатов, статобработка, протоколирование. Собственно исследование (получение

данных) происходит внутри цикла по набору статистики (рис. 1, правый столбец). Внутри цикла выполняются:

– пустая прокрутка (без обработки);

– одноядерная реализация алгоритма (табл. 1);

– многоядерная (в данном случае двухядерная) реализация алгоритма (табл. 2).

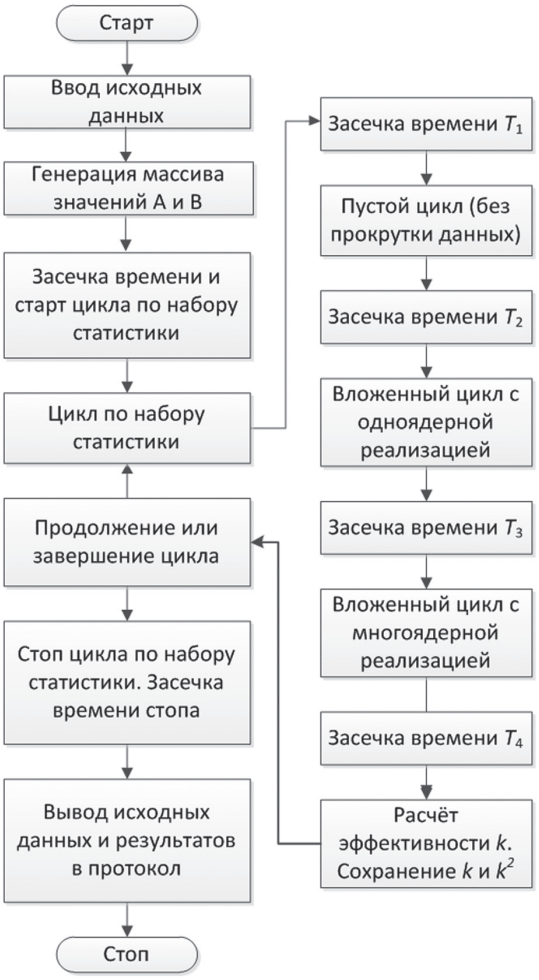


Рис. 1. Блок-схема программы исследования роста эффективности распараллеленного алгоритма по сравнению с последовательным

Число итераций по выполнению указанных трёх процедур является одним из параметров исследования. В частности, в группе данных, представленной ниже на рис. 2, число итераций варьируется в пределах 2 порядков: $10^4 – 10^6$. Перед, после и между процедурами снимаются засечки времени T_1, T_2, T_3, T_4. Время выполнения пустого цикла вычитается из времён выполнения одноядерного и многоядерного циклов. Коэффициент эффективности

$$k = \frac{(T_3 - T_2) - (T_2 - T_1)}{(T_4 - T_3) - (T_2 - T_1)}$$

накапливается в первой степени и в квадрате, для расчёта моментов статистического распределения.

В ходе *предварительных* экспериментов установлено, что распараллеливание одинарной ЛП вычислительной операции не приводит к выигрышу в производительности. Одноядерная реализация ЛП алгоритма выполняется даже несколько быстрее многоядерной. Это обусловлено тем, что выигрыш от распараллеливания теряется на фоне затрат на процедуры выделения конкретных ядер под конкретные операции. Выигрыш появляется, если используются массивы входных данных: распараллеливание вычисления происходит на потоках, привязанных к конкретным ядрам процессора.

С учётом обнаруженной особенности, *окончательный* план эксперимента был скорректирован: в двухъядерном варианте итерации по Шагам 1-3 и 4-7 были выделены в две раздельные группы с сохранением одинакового числа итераций в каждой группе. В исправленном варианте структура распределения ядер процессора задаётся однократно для всей группы данных, а не переназначается всякий раз заново. Таким образом, исчезают потери времени на переназначение, которые в действительности непосредственно не относятся к проведению вычислений в алгоритме. Как и ожидалось, с ростом числа итераций выигрыш от распараллеливания начинает превалировать над потерями времени на выделение ядра.

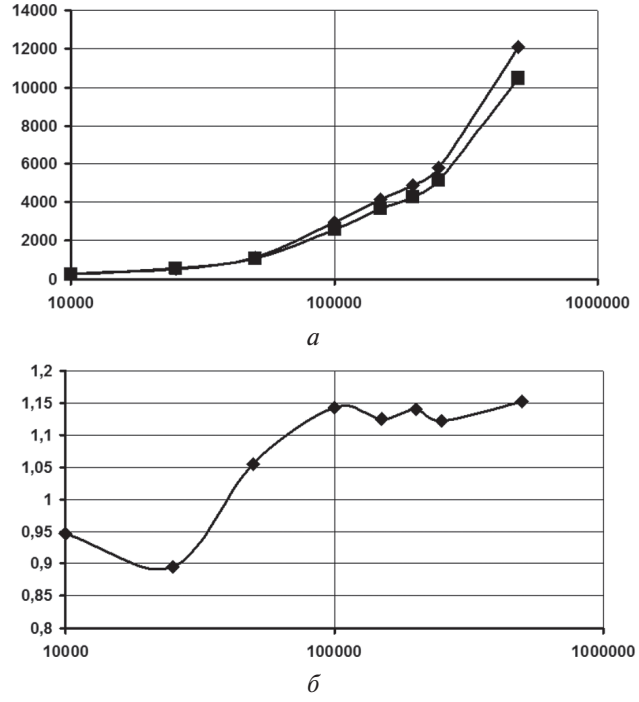
Исследования проведены на аппаратно-программной платформе Intel Pentium Dual-Core E5200; 2.5 GHz; 4 Gb RAM; OS Windows 7 x86. Типовые полученные результаты иллюстрируются графиками (рис. 2 *а* и *б*). На графиках по оси абсцисс отложено число итераций в логарифмических координатах. На рис. 2 *а* представлены зависимости времени выполнения циклов итераций в микросекундах для одноядерной (верхняя кривая) и многоядерной (нижняя кривая) реализаций. Согласно числовым данным на участке до ~30000 итераций график одноядерной кривой проходит ниже графика многоядерной, что отображено зависимостью коэффициента эффективности (рис. 2 *б*). Таким образом, представленный участок изменения числа итераций – в пределах двух порядков – отображает переход от проигрышности к выигрышности по эффективности в точке ~30000 итераций.

Полученные результаты демонстрируют выигрыш эффективности ~15% при циклической (поточной) организации вычислений внутри распараллеленной части алгоритма, что характерно, в частности, при обработке массивово однотипных данных. Как следует из представленного на рис. 2 *б*, при объёме итераций свыше 100000 график эффективность «выходит на полку».

Наличие «полки» на графике указывает на «стабилизацию» зависимости, приближение к асимптоте, являющейся оценкой выигрыша от распараллеливания по сравнению с потерями на однократное распределение потоков по ядрам процессора.

Переход от проигрышности алгоритма к выигрышности может быть охарактеризован как «плавный» или «резкий», в зависимости от масштаба итераций, применительно к конкретной решаемой

задаче. График рис. 2 *б* иллюстрирует изменение числа итераций на 2 порядка и охватывает переход между двумя «полками»: превалирование затрат времени на выделение ядра и выигрыш от распараллеливания вычислений. Таким образом, выигрышность алгоритма критична к динамическому диапазону по числу итераций. Если алгоритм должен применяться к выборкам разного объёма в районе «перехода между двумя полками», то в вопросе эффективности (по выбору между одноядерным и двуядерным вариантами) требуется дополнительное исследование.



а

б

Рис. 2. Зависимости времени выполнения (*а*) и коэффициента эффективности (*б*) от числа итераций в машинном эксперименте

ЛП алгоритм нечёткого теоретико-множественного пересечения (табл. 1 и 2), взятый за основу для оценки эффективности проведения многоядерных ЛП вычислений, является типовым по сложности для алгоритмов нечёткой логики [1, 2]. Как отмечено выше (п. 2), обрабатываются неотрицательные ограниченные величины, интерпретируемые как значения функции принадлежности. Аналогичную сложность имеют ЛП алгоритмы алгебраических нечётких операций. Поэтому полученный результат может рассматриваться как типовой для наработанного класса ЛП алгоритмов операций нечёткой логики.

Интересна так же оценка эффективности для данного класса алгоритмов на матричных операциях. При этом появляются дополнительные параметры машинного эксперимента, связанные с тем, что матрица есть двумерный объект. Соответственно, обработка элементов матрицы в комбинации с ЛП представлением данных может распараллеливаться на большее число ядер.

Выводы

На примере алгоритма локально-параллельной операции нечёткого теоретико-множественного пересечения методом машинного эксперимента показано, что выигрыш в эффективности от использования внутрипроцессорной многоядерности может составлять десятки процентов. С применением стандартного инструментария WinAPI реализован выигрыш ~15% при потоковой обработке данных внутри распараллеленного участка. Потоковая структура приемлема, в частности, при обработке крупных массивов данных, для повышения производительности обработки которых и предназначены локально-параллельные алгоритмы.

Список литературы: 1. *Михаль, О.Ф.* Моделирование распределенных информационно-управляющих систем средствами локально-параллельных алгоритмов обработки нечеткой информации [Текст] / О.Ф. Михаль // Проблемы бионики. Всеукраинский межведомственный научно-технический сборник. – Харьков: ХНУРЭ. – 2001. – Вып. 54. – С. 28-34. **2**. *Михаль, О.Ф.* Принципы организации систем нечеткого регулирования на однородных локально-параллельных алгоритмах [Текст] / О.Ф. Михаль, О.Г. Руденко // Управляющие системы и машины. – 2001. – № 3. – С. 3-10. **3**. *Осотов, И.А.* Применение локально-параллельных алгоритмов с процессорно-зависимой многопоточностью [Текст] / И.А. Осотов, О.Ф. Михаль // Информацион-ные технологии в навигации и управлении: состояние и перспективы развития. Материалы первой международной научно-технической конференции. – К.: ДП "ЦНДІ НіУ", 2010. – С. 53. **4.** *Осотов, И.А.* Реализация локально-параллельных алгоритмов с использованием процессорно-зависимой многопоточности [Текст] / И.А. Осотов, О.Ф. Михаль // Информатика, математическое моделирование, экономика: Сборник научных статей по итогам Международной научно-практической конференции, г. Смоленск, 22 апреля 2011 г. В 2-х томах. Том 1 – Смоленск: Смоленский филиал АНО ВПО ЦС РФ "Российский университет кооперации", 2011. – С. 14-20. **5.** *Милнер, П.* Физиологическая психология [Текст] / П. Милнер. – М.: Мир, 1973. – 648 с. **6**. Википедия. DEC Alpha // (http://ru.wikipedia.org/wiki/ DEC_Alpha).

Поступила в редакцию 5.12.2011

УДК 681.513

Оцінка ефективності проведення локально-паралельних обчислень на багатоядерних процесорах / І.А. Осотов, О.Ф. Міхаль // Біоніка інтелекту : наук.-техн. журнал. – 2012. – № 1 (78). – С. 91-96.

Проведено машинний експеримент щодо оцінки ефективності застосування внутрішньо-процесорної багатоядерної обробки в алгоритмі локально-паралельної операції нечіткого теоретико-множинного перетину. За потокової обробки даних всередині паралеленої частки з застосуванням стандартного інструментарію WinAPI реализовано виграш ефективності порядку 15%.

Табл. 2. Іл. 2. Бібліогр.: 6 найм.

UDK 681.513

Estimation of efficiency of local-parallel calculations based on multikernel processors / I.A. Osotov, O.Ph. Mikhal // Bionics of Intelligense: Sci. Mag. – 2012. – № 1 (78). –P. 91-96.

The machine experiment is organized on estimation of efficiency of multikernel processing of local-parallel algorithm of fuzzy sets intersection operation. Advantage to efficiency is found of the order 15% under stream data processing in parallel area with using standard toolbox WinAPI.

Tabl. 2. Fig. 2. Ref.: 6 items.