УДК 004.852, 004.272.26

ПАРАЛЛЕЛЬНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА ГРАДИЕНТНОГО БУСТИНГА ДЕРЕВЬЕВ РЕШЕНИЙ

П. Н. Дружков[1]

Предлагается программная реализация параллельного алгоритма градиентного бустинга деревьев решений, предполагающего распределенное хранение данных и предназначенного, в первую очередь, для решения больших задач машинного обучения. Приводятся результаты вычислительных экспериментов, показавших преимущество в производительности и масштабируемости предлагаемой программной реализации над доступными открытыми реализациями при использовании выборок больших объемов. Приводятся результаты экспериментальной оценки качества, также показавшие конкурентоспособность предлагаемой реализации. Работа выполнена в рамках программы “Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007–2013 годы” (государственный контракт № 11.519.11.4015) и ФЦП “Научные и научно-педагогические кадры инновационной России на 2009–2013 годы” (государственный контракт № 14.B37.21.0393). Статья рекомендована к публикации Программным комитетом форума “Суперкомпьютерные технологии в образовании, науке и промышленности” (HPC-2012; http://agora.guru.ru/hpc2012).

Ключевые слова: дерево решений, градиентный бустинг, параллельные вычисления, MPI, распределенная память.

1. Введение. Обучение с учителем, или обучение по прецедентам, заключается в восстановлении неизвестной общей зависимости между набором входных переменных (признаков) и некоторым целевым (выходным) показателем на основании конечного набора пар вида “вход-выход”. Чем больше прецедентов доступно на этапе обучения, тем бóльшим объемом информации о восстанавливаемой зависимости располагает алгоритм и, следовательно, может построить более точную модель зависимости. Таким образом, сама возможность применения алгоритма машинного обучения в случае наличия большой обучающей выборки может повысить качество решения задачи. С другой стороны, в последнее время появляются все более и более объемные наборы данных, связанные с важными прикладными задачами, например извлечение информации из поисковых интернет-запросов и интернет-реклама [1], а также интеллектуальный анализ изображений и видео [2]. В связи с этим возникает необходимость модификации существующих алгоритмов или разработки новых, ориентированных именно на обработку больших наборов данных. В настоящей статье представлена открытая параллельная программная реализация на распределенную память одного из наиболее перспективных алгоритмов решения задачи обучения с учителем — градиентного бустинга деревьев решений [3] и производится ее сравнение с другими открытыми реализациями.

Постановка задачи обучения с учителем приведена в разделе 2 данной статьи. Раздел 3 содержит краткое описание алгоритма обучения одиночного дерева решений и градиентного бустинга. Возможные пути параллелизации этих алгоритмов изложены в разделе 4. Раздел 5 посвящен краткому описанию разработанной параллельной программной реализации. Результаты вычислительных экспериментов приведены в разделе 6.

2. Постановка задачи. Одной из задач, рассматриваемых в машинном обучении, является задача обучения с учителем. В рамках этой задачи дано некоторое множество допустимых объектов (входов) X, описываемых векторами признаков. Множество X называют также пространством признаков. Каждому объекту $x \in X$ поставлена в соответствие величина y, называемая выходом (или целевой переменной) и принадлежащая множеству допустимых выходов Y. Упорядоченная пара “вход–выход” (x, y), где $x \in X$ и $y \in Y$, называется прецедентом. Требуется восстановить зависимость между входом и выходом на основе данных о конечном наборе прецедентов, называемом обучающей выборкой: $\left\{(x_i, y_i)\colon x_i \in X, y_i \in Y, i = \overline{1,n}\right\}$. Другими словами, задача состоит в построении функции f из некоторого множества K, которая,

[1] Нижегородский государственный университет им. Н. И. Лобачевского, факультет вычислительной математики и кибернетики, просп. Гагарина, д. 23, 603950, г. Нижний Новгород; аспирант, e-mail: druzhkov.paul@gmail.com

получив на вход x, предсказала бы значение ответа y как можно точнее. В случае конечного Y говорят о задаче классификации; если $Y = \mathbb{R}$, то о задаче восстановления регрессии [4]. Компоненты векторов x, принимающие значения из множества вещественных чисел, называют количественными признаками, а те компоненты, которые принимают значения из конечного неупорядоченного множества, — номинальными или категориальными. Процесс нахождения f называется обучением (тренировкой, настройкой) модели, процесс определения выхода по некоторому входу с помощью уже построенной модели — предсказанием.

3. Деревья решений и бустинг. Одними из наиболее популярных и перспективных в машинном обучении являются алгоритмы, основанные на построении деревьев решений и их ансамблей. Дерево решений строит разбиение пространства признаков на непересекающиеся области с помощью рекурсивной процедуры [5]. Фактически, каждому узлу дерева соответствует некоторая область пространства признаков и правило, по которому осуществляется ее разделение на две приписанных к дочерним вершинам. Вид правила разбиения зависит от типа участвующей в нем переменной: для количественного признака s — это неравенство вида $x^{(s)} < v^{(s)}$, где $x^{(s)}$ — s-я координата вектора входных переменных x, а $v^{(s)}$ — некоторое пороговое значение; для номинального признака правило имеет вид $x^{(s)} \in P$, где P — подмножество множества возможных значений переменной $x^{(s)}$. Поиск оптимального разбиения по количественной переменной производится для всех $v^{(s)} \in V^{(s)} = \left\{ \dfrac{\widehat{x}_j^{(s)} + \widehat{x}_{j+1}^{(s)}}{2} : \widehat{x}_j^{(s)} \leqslant \widehat{x}_{j+1}^{(s)}, j = \overline{1, n-1} \right\}$, где $\widehat{x}_j^{(s)}$ — упорядоченные значения s-го признака. Для обучения дерева решений используется жадная стратегия минимизации функции штрафа $I(D)$, описывающей неоднородность данных в области D, соответствующей некоторому узлу. Типичным выбором для регрессионных деревьев является квадратичная функция $I(D) = \sum\limits_{i:\ x_i \in D} (y_i - \overline{y})^2$. Одним из возможных критериев остановки построения дерева решений является достижение им заданной высоты.

Как уже отмечалось выше, практический интерес представляет не только обучение одиночных деревьев решений, но и построение моделей в виде их ансамблей. Так, бустинг-алгоритмы последовательно строят базовые модели (в настоящей работе — деревья решений) таким образом, что каждая следующая корректирует ошибки предыдущих, тем самым улучшая качество модели. Одним из наиболее общих методов этого направления является алгоритм градиентного бустинга деревьев решений [3], который использует аналогию с методом градиентного спуска для построения ансамбля, минимизирующего ошибку на обучающей выборке, заданную дифференцируемой неотрицательной функцией потерь (штрафа) $L(y, y')$. Этот алгоритм строит модель в виде суммы деревьев $f_M(x) = h_0 + \nu \sum\limits_{j=1}^{M} h_j(x)$, где h_0 — некоторая константная модель (начальное приближение), $\nu \in (0, 1]$ — параметр, регулирующий скорость обучения, $h_j(x)$ — регрессионные деревья решений. Слагаемые-деревья добавляются в сумму последовательно так, чтобы очередное дерево решений наилучшим образом предсказывало не исходные значения целевой переменной, а антиградиент функции потерь $-\left[\dfrac{\partial L\big(y_i, F(x_i)\big)}{\partial F(x_i)} \right]_{F = f_{m-1}}$, зависящий от истинных значений выходного признака и предсказаний модели, построенной к текущей итерации. Следует отметить, что каждое дерево решений обучается с использованием квадратичного штрафа, что не требует модификации алгоритма его построения.

Поскольку добавление деревьев решений в модель градиентного бустинга должно производиться последовательно, параллелизм возможен лишь на уровне одиночных деревьев решений.

4. Параллельное обучение дерева решений. Очевидным путем распараллеливания процесса обучения дерева решений является одновременное построение разбиений в нескольких узлах дерева, находящихся на одинаковой глубине, или непересекающихся поддеревьев, однако такой подход не позволяет добиться хорошей масштабируемости для деревьев решений малой высоты, характерных для бустинг-алгоритмов. Более эффективные способы основаны на распределении данных между вычислителями: двумя конкурирующими подходами являются “вертикальное” и “горизонтальное” разделение.

При вертикальном разделении между вычислителями (процессами, потоками) распределяются признаки, и в каждом узле дерева поиск наилучших правил разбиения выполняется параллельно по различным признакам. Основные трудности реализации этого подхода на системы с распределенной памятью связаны с тем, что после построения оптимального разбивающего правила в некотором узле дерева лишь один вычислитель содержит информацию о том, в какой именно дочерний узел отправятся те или иные прецеденты обучающей выборки. Данное обстоятельство требует эффективных методов передачи этой

информации всем процессам.

При горизонтальном разделении между вычислителями распределяются прецеденты обучающей выборки. Следует отметить, что поскольку каждому отдельно взятому процессу не доступна вся выборка, вычислить все элементы множества $V^{(s)}$ для количественного признака s не представляется возможным при сохранении потенциальной эффективности этого подхода. В связи с этим, в [1] предлагается перед процессом обучения вычислить для каждой количественной переменной элементы множества $V^{(s)}$ как оценки квантилей уровней $\frac{i}{q}$, $i = \overline{1, q-1}$, для относительно небольшого фиксированного числа q. Такая модификация позволяет каждому процессу независимо от остальных оценить влияние доступных ему прецедентов на качество допустимых разбиений с помощью построения гистограмм, которые затем суммируются на одном процессе для выбора оптимального правила разбиения. Схожий подход рассматривается в [6], однако множество $V^{(s)}$ не фиксируется перед обучением дерева, а формируется динамически при построении каждого правила разбиения. Следует отметить, что в связи с ограничением множества рассматриваемых правил разбиения по количественным признакам рассмотренные параллельные модификации не эквивалентны исходному алгоритму обучения дерева решений, что может сказаться на качестве получаемых моделей, однако экспериментальные данные показывают, что если ухудшение качества и наблюдается, то оно, как правило, незначительно [1, 6].

5. Параллельная программная реализация градиентного бустинга деревьев решений. На настоящий момент за редким исключением существует проблема доступности программных реализаций описанных выше подходов. В связи с этим была создана программная реализация с открытым исходным кодом [7] алгоритма обучения одиночного дерева решений и градиентного бустинга деревьев решений, основанных на горизонтальном распределении данных в соответствии с [1]. Для градиентного бустинга были реализованы следующие функции потерь для задач восстановления регрессии: абсолютная, квадратичная, функция Хьюбера, а для классификации — логистическая и кросс-энтропия [3]. Эта реализация ориентирована на использование на HPC-кластерах, где для обеспечения параллелизма используется технология MPI. Распределение данных производится между рабочими процессами, в то время как один управляющий процесс выделен для сбора информации с рабочих процессов и для определения оптимальных правил разбиения на основе полученных гистограмм.

6. Вычислительный эксперимент. Как отмечалось выше, представляемые в литературе параллельные модификации рассматриваемых алгоритмов, как правило, не имеют доступных программных реализаций, что затрудняет объективное сравнение существующих подходов. Кроме того, реализации различных алгоритмов ориентированы на использование разных моделей распределенных вычислений (MPI, MapReduce и т.д.) и тестируются авторами на разных наборах данных.

В настоящей работе проводится сравнение предлагаемой программной реализации градиентного бустинга (MPIGBT), основанной на горизонтальном распределении данных, с двумя алгоритмами вертикального разделения: открытой кластерной реализацией (PGBRT) [8] и реализацией на общую память (CvGBTrees) [9–11]. Рассматривается задача упорядочивания выдаваемых интернет-поисковиком результатов по релевантности. Одним из возможных подходов к решению данной задачи является ее сведéние к регрессионной задаче предсказания числовой характеристики релевантности каждого документа в отдельности [6]. Для построения моделей градиентного бустинга используются наборы данных Yahoo Learning to Rank Challenge 2010 [12] и Microsoft MSLR-WEB30k [13], описание которых приведено в таблице.

Характеристики используемых наборов данных

Набор данных	Yahoo LTRC Set 1	Yahoo LTRC Set 2	MSLR Fold 1
Количество признаков	700	700	136
Объем обучающей выборки	473134	34815	$S_1 + S_2 + S_3 = $ $= 756149 + 753394 + 760753$
Объем тестовой выборки	165660	103174	753611

Все эксперименты проводились на вычислительном кластере МГУ “Ломоносов” [14]. Для создания исполняемых файлов использовался компилятор C++ Intel 13.1.0, в качестве библиотеки обмена сообщениями — Intel MPI 4.1.0.

Была проведена серия экспериментов с целью изучения масштабируемости рассматриваемых реализаций при использовании различных наборов данных. При обучении модели градиентного бустинга строилось 250 деревьев, с максимальной глубиной каждого из них 4, параметр скорости обучения $\nu = 0.1$,

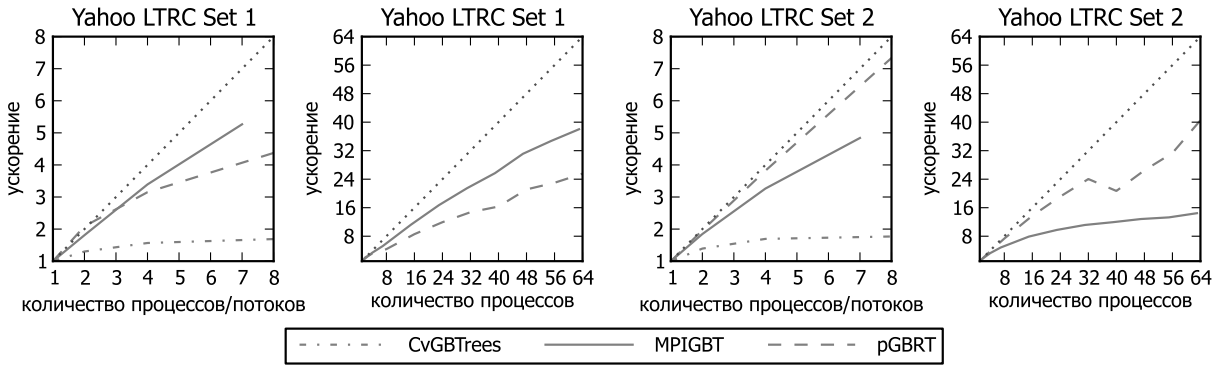

Рис. 1. Масштабируемость различных программных реализаций обучения градиентного бустинга деревьев решений на данных Yahoo LTRC относительно используемых вычислительных ресурсов

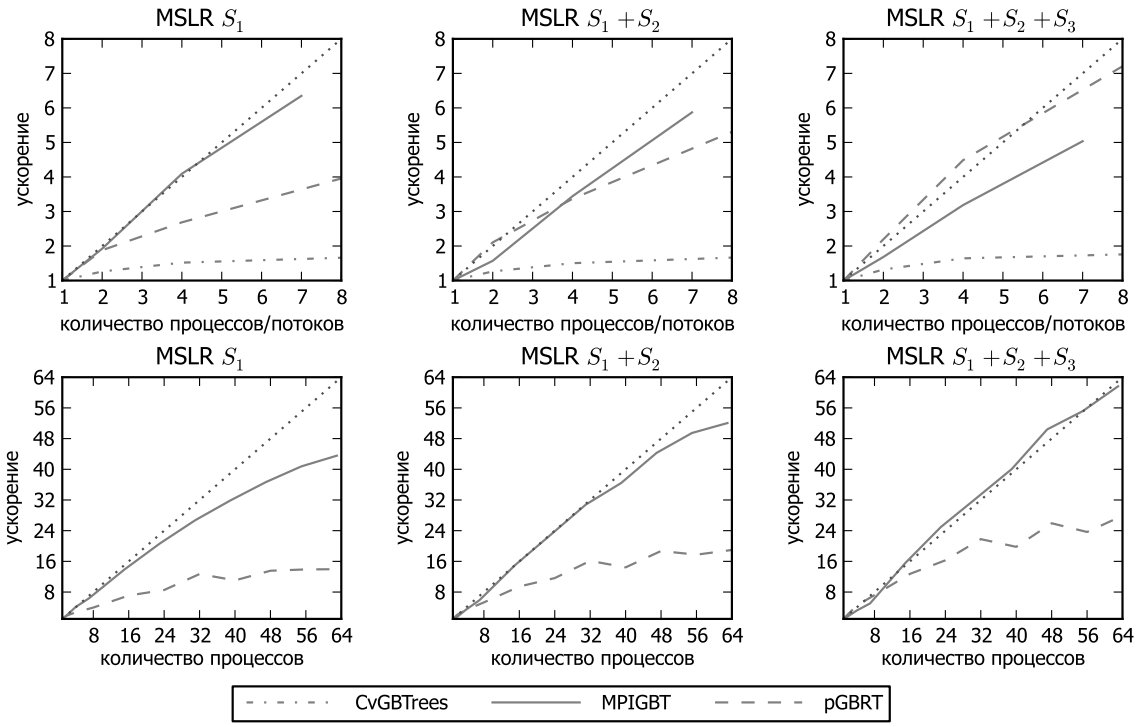

Рис. 2. Масштабируемость различных программных реализаций обучения градиентного бустинга деревьев решений на данных Microsoft MSLR-WEB30k относительно используемых вычислительных ресурсов

размер гистограмм для MPIGBT равнялся 100. В связи с тем, что реализация [8] поддерживает лишь квадратичную функцию потерь, только она используется в описанных ниже экспериментах. Масштабирование реализации MPIGBT измерялось по отношению к увеличению количества рабочих процессов (общее количество процессов на единицу больше). На рис. 1 представлены кривые масштабируемости на данных Yahoo при использовании как одного вычислительного узла кластера (число потоков/процессов до 8), так и нескольких (до 64 процессов).

Использование одного вычислительного узла позволяет сделать сравнение реализаций для общей и распределенной памяти. Как видно из приведенных графиков, реализации MPIGBT и PGBRT обеспечивают значительно бóльшие ускорения по сравнению с CvGBTrees, что еще раз подчеркивает значимость аккуратного распараллеливания по признакам. При этом время работы реализации CvGBTrees на рассматриваемых наборах данных превосходит время работы как MPIGBT, так и PGBRT. Отставание реализации MPIGBT от PGBRT на наборе данных Yahoo LTRC Set 2 можно объяснить тем, что объем обучающей выборки в нем слишком мал для эффективного использования горизонтального распределения данных, в то время как достаточно большое количество признаков способствует относительно

хорошей масштабируемости реализации PGBRT. Можно видеть, что тенденции, замеченные на одном вычислительном узле, сохраняются и при большем количестве используемых узлов кластера. Из экспериментальных результатов видно, что увеличение объема выборки позитивно влияет на масштабируемость реализации MPIGBT, что еще заметней на наборе данных Microsoft (рис. 2).

Обучающая выборка данных Microsoft разделена на три приблизительно равные части S_1, S_2 и S_3. Масштабируемость рассматриваемых реализаций была исследована сначала на одной части S_1, затем на двух $S_1 + S_2$ и, наконец, на всей доступной обучающей выборке $S_1 + S_2 + S_3$. Полученные результаты еще раз подчеркивают, что большие наборы данных наилучшим образом подходят для реализации MPIGBT. Следует отметить, что рост объема выборки приводит и к увеличению ускорений работы PGBRT, однако в меньшей степени. Кроме того, меньшее количество признаков также не лучшим образом подходит для реализации PGBRT.

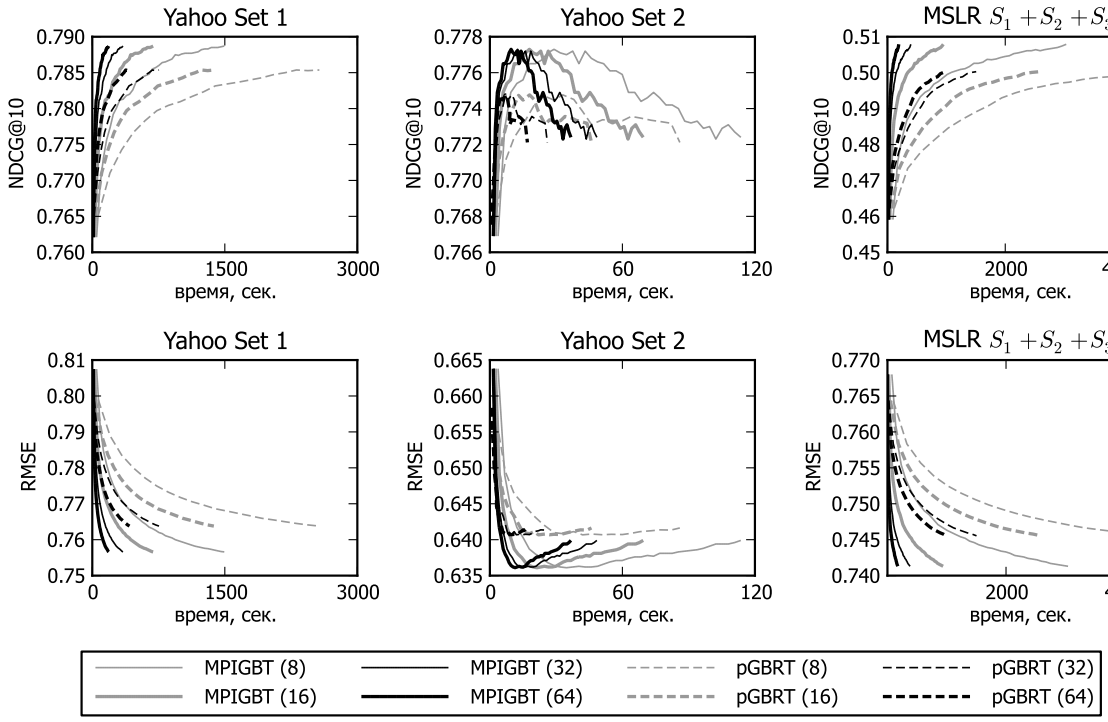

Рис. 3. Зависимость качества обучаемых моделей, оцененного по тестовой выборке, от времени работы алгоритма при различном объеме вычислительных ресурсов (8, 16, 32 и 64 MPI-процесса)

Несмотря на лучшие достигнутые на рассматриваемых наборах данных показатели ускорения, реализация MPIGBT потенциально может строить модели с более низким качеством, так как лежащий в ее основе алгоритм обучения дерева решений рассматривает суженное множество допустимых разбиений в узлах деревьев. В связи с этим, были проведены эксперименты, в ходе которых измерялось качество получаемой модели на тестовой выборке с помощью популярного показателя качества ранжирования — NDCG, вычисляемого по методологии Yahoo [15]. Поскольку в данной работе задача ранжирования сводится к задаче восстановления регрессии, использовалась распространенная метрика оценки качества решения регрессионных задач — корень среднеквадратической ошибки (RMSE) предсказания числовых показателей релевантности документов.

В рамках обсуждаемой серии экспериментов использовались те же параметры алгоритмов обучения, что и ранее, за исключением количества бустинг-итераций. Полученные результаты приведены на рис. 3 в виде графиков зависимости качества получаемых моделей от времени работы соответствующего алгоритма. Из полученных результатов можно видеть, что реализация MPIGBT позволяет получить модель сопоставимого с PGBRT-моделью качества за значительно меньшее время при использовании выборок большого объема. При этом на самом маленьком наборе данных Yahoo LTRC Set 2 время работы обеих реализаций сравнимо, однако при этом эффект переобучения (увеличение тестовой ошибки при уменьшении ошибки обучения) [4] для алгоритма PGBRT возникает раньше, не позволяя получить качество, аналогичное MPIGBT-модели.

7. Заключение. В настоящей статье были рассмотрены некоторые подходы к распараллеливанию алгоритма обучения деревьев решений и градиентного бустинга деревьев решений, представлена программная реализация с открытым исходным кодом одного из алгоритмов для систем с распределенной памятью. Приведены результаты вычислительных экспериментов сравнения масштабируемости представленной реализации со свободно распространяемыми реализациями конкурирующего подхода к распараллеливанию бустинг-алгоритмов на общую и распределенную память. Результаты показали, что масштабируемость алгоритма, основанного на горизонтальном распределении данных между процессами, ожидаемо улучшается с ростом объема обучающей выборки, в то время как вертикальное распределение оказывается предпочтительнее на меньшей выборке. Результаты оценки качества получаемых моделей свидетельствуют о том, что предлагаемая реализация позволяет за меньшее время обучать модели, по качеству не уступающие получаемым с помощью исходного алгоритма градиентного бустинга.

Автор благодарит Н. Ю. Золотых и И. Б. Меерова за полезные обсуждения и внимание к работе, а также выражает благодарность рецензенту, указавшему на многие недостатки первого варианта рукописи.

СПИСОК ЛИТЕРАТУРЫ

1. *Panda B., Herbach J.S., Basu S., Bayardo R.J.* PLANET: massively parallel learning of tree ensembles with MapReduce // Proc. of the 35th International Conference on Very Large Data Base (VLDB). New York: ACM Press, 2009. 1426–1437.
2. *Dollar P., Wojek C., Schiele B., Perona P.* Pedestrian detection: a benchmark // Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPRT09). New York: IEEE Press, 2009. 304–311.
3. *Friedman J.* Greedy function approximation: the gradient boosting machine // Annals of Statistics. 2001. **29**, N 5. 1189–1232.
4. *Hastie T., Tibshirani R., Friedman J.* The elements of statistical learning: data mining, inference, and prediction. New York: Springer, 2009.
5. *Breiman L., Friedman J.H., Olshen R.A., Stone C.J.* Classification and regression trees. Belmont: Wadsworth, 1984.
6. *Tyree S., Weinberger K.Q., Agrawal K., Paykin J.* Parallel boosted regression trees for web search ranking // Proc. of the 20th Int. Conf. on World Wide Web (WWW). New York: ACM Press, 2011. 387–396.
7. MPIGBT. Параллельная реализация алгоритма обучения градиентного бустинга деревьев решений на распределенную память (http://ml.vmk.unn.ru/index.php/ru/resources-ru).
8. pGBRT: Parallel Gradient Boosted Regression Trees (http://machinelearning.wustl.edu/pmwiki.php/Main/Pgbrt).
9. *Druzhkov P.N., Eruhimov V.L., Kozinov E.A., Kustikova V.D., Meyerov I.B., Polovinkin A.N., Zolotykh N.Yu.* On some new object detection features in OpenCV Library // Pattern Recognition and Image Analysis. 2011. **21**, N 2. 377–379.
10. *Дружков П.Н., Золотых Н.Ю., Половинкин А.Н.* Программная реализация алгоритма градиентного бустинга деревьев решений // Вестн. Нижегородского гос. ун-та им. Н. И. Лобачевского. 2011. № 1. 193–200.
11. *Дружков П.Н., Золотых Н.Ю., Половинкин А.Н.* Реализация параллельного алгоритма предсказания в методе градиентного бустинга деревьев решений // Вестн. Южно-Уральского гос. ун-та. Серия: Математическое моделирование и программирование. 2011. № 37. 82–89.
12. *Chapelle O., Chang Y.* Yahoo! learning to rank challenge overview // J. of Machine Learning Research. 2011. N 14. 1–24.
13. Microsoft Learning to Rank Datasets (http://research.microsoft.com/en-us/projects/mslr).
14. *Воеводин Вл.В., Жуматий С.А., Соболев С.И., Антонов А.С., Брызгалов П.А., Никитенко Д.А., Стефанов К.С., Воеводин Вад.В.* Практика суперкомпьютера "Ломоносов" // Открытые системы. 2012. № 7. 36–39.
15. *Busa-Fekete R., Szarvas Gy., Elteto T., Kegl B.* An apple-to-apple comparison of learning-to-rank algorithms in terms of normalized discounted cumulative gain // Proc. of ECAI-12 Workshop on Preference Learning: Problems and Applications in AI. 2012 (http://www.ke.tu-darmstadt.de/events/PL-12/papers/07-busa-fekete.pdf).

Поступила в редакцию
01.04.2013