Information Processing Letters 114 (2014) 264–267

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

Fast monotone summation over disjoint sets ☆

Petteri Kaski [a], Mikko Koivisto [b], Janne H. Korhonen [b,*], Igor S. Sergeev [c]

[a] Helsinki Institute for Information Technology HIIT & Department of Information and Computer Science, Aalto University, PO Box 15400, FI-00076 Aalto, Finland
[b] Helsinki Institute for Information Technology HIIT & Department of Computer Science, University of Helsinki, PO Box 68, FI-00014 Helsinki, Finland
[c] Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Department of Discrete Mathematics, Leninskie Gory, Moscow 119991, Russia

ARTICLE INFO

Article history:
Received 8 March 2013
Accepted 6 December 2013
Available online 10 December 2013
Communicated by J. Torán

Keywords:
Algorithms
Arithmetic complexity
Commutative semigroup
Disjoint summation
k-Path
Matrix permanent

1. Introduction

Let $(S,+)$ be a commutative semigroup and let $[n]=\{1,2,\ldots,n\}$. For integers $0 \leqslant p,q \leqslant n$, the (n,p,q)-*disjoint summation* problem is as follows. Given a value $f(X) \in S$ for each set $X \subseteq [n]$ of size at most p, the task is to output the function e, defined for each set $Y \subseteq [n]$ of size at most q by

$$e(Y) = \sum_{X \cap Y = \emptyset} f(X), \tag{1}$$

where $X \subseteq [n]$ ranges over sets of size at most p that are disjoint from Y.

We study the arithmetic complexity [6] of (n,p,q)-disjoint summation, with the objective of quantifying how many binary additions in S are sufficient to evaluate (1) for all Y. As additive inverses need not exist in S, this is equivalent to the monotone arithmetic circuit complexity, or equivalently, the monotone arithmetic straight-line program complexity of (n,p,q)-disjoint summation.

Our main result is the following. Let $C_{n,p,q}$ denote the minimum number of binary additions in S sufficient to compute (n,p,q)-disjoint summation, and let us write $\binom{n}{\downarrow k}$ for the sum $\binom{n}{0}+\binom{n}{1}+\cdots+\binom{n}{k}$ of binomial coefficients.

Theorem 1. $C_{n,p,q} \leqslant \left[p\binom{n}{\downarrow p}+q\binom{n}{\downarrow q}\right] \cdot \min\{2^p,2^q\}$.

This improves upon a preliminary result by a subset of the present authors [10].

2. Related work and applications

Circuit complexity. If we ignore the empty set, the special case of $(n,1,1)$-disjoint summation corresponds to

☆ This paper is an extended version of a conference abstract by a subset of the present authors [10].

* Corresponding author.
E-mail addresses: petteri.kaski@aalto.fi (P. Kaski), mikko.koivisto@cs.helsinki.fi (M. Koivisto), janne.h.korhonen@cs.helsinki.fi (J.H. Korhonen), isserg@gmail.com (I.S. Sergeev).

0020-0190/$ – see front matter
http://dx.doi.org/10.1016/j.ipl.2013.12.003

the matrix–vector multiplication task $x \mapsto \bar{I}x$, where $\bar{I}$ is the complement of the identity matrix, that is, a matrix with zeroes on diagonal and ones elsewhere. Results from Boolean circuit complexity concerning this particular task can be transferred to our setting, implying that $(n,1,1)$-disjoint summation has complexity exactly $3n-6$ [7,14].

For the special case of (n,n,n)-disjoint summation, Yates [18] gave an algorithm evaluating (1) for all $Y \subseteq [n]$ using $2^{n-1}n$ additions in S; Knuth [12, §4.6.4] presents a modern exposition. Furthermore, this bound is tight in the monotone setting [11]. Yates's algorithm has been used as a component in $2^n n^{O(1)}$ time algorithms for graph k-colouring and related covering and packing problems [3].

In a setting where S is a group and we are allowed to take additive inverses in S, it is known that the (n,p,q)-disjoint summation can be evaluated with $O(p\binom{n}{\downarrow p} + q\binom{n}{\downarrow q})$ operations in S via the principle of inclusion and exclusion. This has been employed to obtain improved counting algorithms for hard combinatorial problems, most prominently the $\binom{n}{k/2}n^{O(1)}$ algorithm for counting k-paths in a graph by Björklund et al. [4].

Maximisation. An immediate template for applications of (n,p,q)-disjoint summation is maximisation under constraints. If we choose the semigroup to be $(\mathbb{Z} \cup \{-\infty,\infty\}, \max)$, then (1) becomes

$$e(Y) = \max_{X \cap Y = \emptyset} f(X).$$

This can be seen as precomputing the optimal p-subset X when the elements in a q-subset Y are "forbidden", in a setting where the set Y is either not known beforehand or Y will eventually go through almost all possible choices. Such scenario takes place e.g. in a recent work [8] on Bayesian network structure learning by branch and bound.

Semirings. Other applications of disjoint summation are enabled by extending the semigroup by a multiplication operation that distributes over addition. That is, we work over a semiring $(S,+,\cdot)$, and the task is to evaluate the sum

$$\sum_{X \cap Y = \emptyset} f(X) \cdot g(Y), \tag{2}$$

where X and Y range over all disjoint pairs of subsets of $[n]$, of size at most p and q, respectively, and f and g are given mappings to S. We observe that the sum (2) equals $\sum_Y e(Y) \cdot g(Y)$, where Y ranges over all subsets of $[n]$ of size at most q and e is as in (1). Thus, an efficient way to compute e results in an efficient way to evaluate (2).

Counting k-paths. An application of the semiring setup is counting the maximum-weight k-edge paths from vertex s to vertex t in a given graph with real edge weights. Here we assume that we are only allowed to add and compare real numbers and these operations take constant time (cf. [16]). By straightforward Bellman–Held–Karp type dynamic programming [1,2,9] we can solve the problem in $\binom{n}{k}n^{O(1)}$ time. Our main result gives an improved algorithm that runs in time $2^{k/2}\binom{n}{k/2}n^{O(1)}$ for even k. The key idea is to solve the problem in halves. We guess a middle vertex v and define $w_1(X)$ as the maximum weight of a $k/2$-edge path from s to v in the graph induced by the vertex set $X \cup \{v\}$; we similarly define $w_2(X)$ for the $k/2$-edge paths from v to t. Furthermore, we define $c_1(X)$ and $c_2(X)$ as the respective numbers of paths of weight $w_1(X)$ and $w_2(X)$ and put $f(X) = (c_1(X), w_1(X))$ and $g(X) = (c_2(X), w_2(X))$. These values can be computed for all vertex subsets X of size $k/2$ in $\binom{n}{k/2}n^{O(1)}$ time. Now the expression (2) equals the number of k-edge paths from s to t with middle vertex v, when we define the semiring operations $\boxplus$ and $\boxdot$ in the following manner: $(c,w) \boxdot (c',w') = (c \cdot c', w + w')$ and

$$(c,w) \boxplus (c',w') = \begin{cases} (c,w) & \text{if } w > w', \\ (c',w') & \text{if } w < w', \\ (c+c',w) & \text{if } w = w'. \end{cases}$$

For the more general problem of counting weighted subgraphs Vassilevska and Williams [15] give an algorithm whose running time, when applied to k-paths, is $O(n^{\omega k/3}) + n^{2k/3+O(1)}$, where $2 \leqslant \omega < 2.3727$ is the exponent of matrix multiplication [17].

Computing matrix permanent. A further application is the computation of the permanent of a $k \times n$ matrix (a_{ij}) over a noncommutative semiring, with $k \leqslant n$ an even integer, given by $\sum_\sigma a_{1\sigma(1)}a_{2\sigma(2)} \cdots a_{k\sigma(k)}$, where the sum is over all injective mappings σ from $[k]$ to $[n]$. We observe that the expression (2) equals the permanent if we let $p = q = k/2 = \ell$ and define $f(X)$ as the sum of $a_{1\sigma(1)}a_{2\sigma(2)} \cdots a_{\ell\sigma(\ell)}$ over all injective mappings σ from $\{1,2,\ldots,\ell\}$ to X and, similarly, $g(Y)$ as the sum of $a_{\ell+1\sigma(\ell+1)}a_{\ell+2\sigma(\ell+2)} \cdots a_{k\sigma(k)}$ over all injective mappings σ from $\{\ell+1,\ell+2,\ldots,k\}$ to Y. Since the values $f(X)$ and $g(Y)$ for all relevant X and Y can be computed by dynamic programming with $O(k\binom{n}{\downarrow k/2})$ operations in S, our main result yields an upper bound of $O(2^{k/2}k\binom{n}{\downarrow k/2})$ operations in S for computing the permanent.

Thus we improve significantly upon a Bellman–Held–Karp type dynamic programming algorithm that computes the permanent with $O(k\binom{n}{\downarrow k})$ operations in S, the best previous upper bound we are aware of for noncommutative semirings [5]. It should be noted, however, that algorithms using $O(k\binom{n}{\downarrow k/2})$ operations in S are already known for noncommutative rings [5], and that faster algorithms using $O(k(n-k+1)2^k)$ operations in S are known for commutative semirings [5,13].

3. Evaluation of disjoint sums

Overview. In this section, we prove Theorem 1 by giving an inductive construction for the evaluation of (n,p,q)-disjoint summation. That is, we reduce (n,p,q)-disjoint summation into one $(n-1,p,q)$-disjoint summation and two $(n-2,p-1,q-1)$-disjoint summations. The key idea is to "pack" two elements of the ground set $[n]$ (say, 1 and n) into a new element $*$ and apply $(n-1,p,q)$-disjoint summation. We then complete this to (n,p,q)-disjoint summation using the two $(n-1,p-2,q-2)$-disjoint summations.