Vatutin E.I., Zaikin O.S., Zhuravlev A.D., Manzuk M.O., Kochemazov S.E., Titov V.S.

Using grid systems for enumerating combinatorial objects on example of diagonal Latin squares

One of important classes of combinatorial and discrete optimization problems [1] is formed by enumeration problems. During their decision one needs to determine a number of objects with specified properties. The simplest examples of such problems are well known problems about chess rooks, chess queens, etc. For some of them precise analytic decisions are known, while to solve others one needs to perform exhaustive search to enumerate the number of decisions satisfying the specified constraints. For example, for chess rooks problem the number of possible dispositions of N rooks on the board of size $N\times N$ matches the number of permutations and is equal to $N!$. For some problems from the considered class the number of decisions can be expressed using Stirling numbers (of the first and second kind), Bell numbers [2], the number of combinations or partial permutations and so on. At the same time the precise analytical formulas for the number of decisions for chess queens problem or the number of Latin squares of order N are unknown (in the latter case there are known upper and lower bounds).

The number of decisions usually grows rapidly with the increase of the dimension of a problem N, that is why when enumerating corresponding objects using brute force strategy one has to develop highly effective program implementation that takes into account the features of considered problem and provides high rate of generation of enumerated objects. From the point of view of parallel programming the enumeration problems of such type are weakly coupled problems, thus the algorithms for their solving can be implemented in the form of parallel programs that are efficient within the context of parallel computing environments with various architecture that comprise grid systems.

One of the combinatorial objects of considered type is diagonal Latin squares (DLS) that are square tables of size $N\times N$, where each cell is filled by an element of some alphabet (typically a number from 0 to N–1) and in each row, each column and also in main and second diagonals all elements are distinct. Basically, DLS are a special case of Latin squares (LS) that satisfy additional diagonality constraints. Using simple transformations that do not violate any of the constraints any DLS can be reduced to DLS in which the elements of the first row are sorted in ascending order. The corresponding squares form an isomorphism class of size $N!$. The dependence of number of LS on N is well known and presented by A000315 sequence in the Online Encyclopedia of Integer Sequences (OEIS) [3], the dependence of the number of LS with fixed first row has the number A000479. For DLS similar sequences are unknown and can be calculated using brute force.

Native program implementation of this enumerating process is rather ineffective and has the rate of generating squares of order 10 less than 1 DLS/s. In order to increase this rate and, as a result, reduce computing time costs we introduced into the implementation the following optimizations: altering the order of filling elements of DLS; using static data structures instead of placing it in the dynamic memory; using information about the number of possible values $\left|S_{ij}\right|$ for none filled cells of square combined with filling the cells with $\left|S_{ij}\right|=1$ out of order and avoiding unpromising branches of combinatorial tree with $\left|S_{ij}\right|=0$ early; using auxiliary data structures (one dimension arrays) for filling the set of allowed items S_{ij} fast; switching off the Hyper-Threading technology during single threaded generation of DLS combined with avoiding background load on the CPU cores not used for generation; selecting the order of the filling cells by criterion $\left|S_{ij}\right|\to\min$ to decrease the arity of nodes of combinatorial tree; using PGO compilation. As a result it was possible to achieve the rate of generation of about 220 000 DLS/s for recurrent single threaded CPU-oriented program implementation on Delphi language and 240 000 DLS/s for similar implementation on C language (processor Intel Core i7 4770). By developing an alternative special iterative program implementation with N^2 nested loops the rate of generation was additionally increased to 790 000 DLS/s.

Thus the developed program implementation is almost 6 orders more effective compared to native implementation, and this fact makes it possible to use it to enumerate some combinatorial objects (for example, DLS and pairs of orthogonal DLS, also known as Graeco-Latin squares). We applied it to enumerating DLS with fixed first row for several values of *N*. The corresponding numerical sequence is as follows: 1, 0, 0, 2, 8, 128, 171200, 7447587840. The total number of DLS can be calculated from the given sequence by multiplying its members by the cardinality of corresponding isomorphism class that is equal to *N*!: 1, 0, 0, 48, 960, 92160, 862848000, 300286741708800. At this moment authors are preparing the computational experiment aimed at organizing distributed enumeration of the number of DLS for cases with greater values of *N* using grid systems organized on voluntary basis.

The research was partially supported by the state assignments for the Southwest State University (2014–2017 years, no. 2246), by Russian Foundation for Basic Research (grants 14-07-00403-a, 15-07-07891-a and 16-07-00155-a) and by Council for Grants of the President of the Russian Federation (grants NSh-8081.2016.9, MK-9445.2016.8 and stipend SP-1184.2015.5). We thank citerra [Russia team] for his help in development and implementing of some algorithms.

Bibliography

1. Vatutin E.I., Titov V.S., Emelyanov S.G. Basics of discrete combinatorial optimization (in Russian). M.: ARGAMAC-MEDIA, 2016. 270 p.
2. Vatutin E.I. Logic multicontrollers design. Getting separations of parallel graph-schemes of algorithms (in Russian). Saarbrucken: Lambert Academic Publishing, 2011. 292 p.
3. https://oeis.org/A000315