An algorithm for determining the optimal variant of a cut gem with maximal mass and specified symmetry deviations[1]

Denis S. Kokorev
Doctoral Student, Laboratory of Distributed Computational Systems
Institute for Information Transmission Problems RAS
Address: 19, build.1, Bolshoy Karetny Per., Moscow, 127051, Russian Federation
E-mail: korvin-d@yandex.ru

Abstract

The article discusses the problem of finding a polyhedron given shape inside another nonconvex polyhedron. This problem is a particular case of the 18th Hilbert problem, third part. It has a practical application in computer simulation of three-dimensional objects, moving autonomous robots, and the jewelry industry. The author uses this mathematical problem to find the facets of gemstones in uncut stones.

The article offers a method for finding inscribed polyhedrons based on the reduction of the problem to a nonlinear programming problem and its solutions using ready-made software. The basic idea is that it is easy to describe this problem in terms of non-linear programming. Internal polyhedron volume is an objective function. Restrictions include the preservation of the combinatorial structure, one polyhedron standing inside another one, convexity, plus additional constraints necessary for practical purposes.

The article describes two implementations of the algorithm: a client-server application and a local application. Their advantages and disadvantages are discussed. The algorithm is described not only in a mathematical point of view; some of its practical characteristics are also demonstrated. Compared to the previous article, the author has added a method that allows for solving the nonconvex case of a problem. This is a significant step forward from a mathematical point of view. In addition, it allows us to use the algorithm at all stages of gem cutting. The end of the article describes current evaluations of the effectiveness and running time, including on weak processors, and it offers plans for further development of the algorithm.

Key words: convex polyhedrons, combinatorial structure, inscribed polyhedron, cut gem, symmetry deviation, nonlinear programming problems, solver.

Citation: Kokorev D.S. (2017) An algorithm for determining the optimal variant of a cut gem with maximal mass and specified symmetry deviations. *Business Informatics*, no. 2 (40), pp. 40–46.
DOI: 10.17323/1998-0663.2017.2.40.46.

Introduction

One of the classical mathematical problems is the third part of Hilbert's eighteenth problem. In its general formulation, it concerns packing of shapes within other shapes. For the case of packing regular shapes, e.g. sphere packing, proven results exist, among which are the following: the strong thirteen spheres problem [1], the 25 spheres problem [2] and the Kepler conjecture [3]. For the case of packing 2-dimensional polyhedrons and simple 3-dimensional shapes, there are a number of solved problems. Some of these problems are listed in the article by Valiakhmetova and Filippova [4]. In the case of packing multi-dimensional polyhedrons within other polyhedrons, the problem be-

[1] This research was supported by the Russian Scientific Foundation (project No. 16-11-10352)

comes too complicated, and there is no theoretical approach to solving the problem in its general form.

One of the special cases of the problem is the problem of finding a body of specified shape of the largest volume within a nonconvex arbitrarily shaped polyhedron. This problem has a wide range of applications: computer modeling of 3-dimensional shapes; programming simulators; computer games; applications to solve packing and cutting stock problems; robot movement programs; the jewelry industry; the processing of expensive materials. The algorithm described in this article is used for commercial purposes to build an optimal plan for a gem cutter during different stages of gemstone processing.

The first step of the algorithm is to find the starting point, i.e. the approximate starting position of the body being packed. The results of other algorithms may be taken as the starting point. The next step is to gradually shift the vertices of the initial shape by a not very big specified distance and find such positions that result in the largest volume of the body and satisfy a number of conditions previously set on the body's shape.

Currently there are several software products for gem cutters which are used to calculate planning of gemstone cuts. However, all of them work with a finite number of rather strictly symmetrical shapes. The results of their algorithms serve as starting points for the algorithm from this article which distorts the symmetry of the initial shape in the limits allowed by international and manufacturing standards to achieve a larger mass and a higher value of the cut gem.

The algorithm has been implemented as a client-server application for research purposes and as a local application for commercial purposes. For the client-server implementation, external software resources have been used which are organized as RESTful web services [5]. This approach allows us not to spend a lot of time on developing complicated mathematical algorithms instead of using ready solutions and, therefore, focusing on the task at hand [6]. For the terms used in this paper, refer to the article [7].

1. Client-server application

The client-server application has four main steps: input data processing, non-linear problem (NLP) formulation, building the polyhedron from a derived solution, solver calculations (Figure 1). The first step is to transform the input data that might be in different formats into the format necessary for further calculations. The second step is to formulate the initial geometric problem as a problem in terms of nonlinear programming (NLP). After that, the nonlinear solver solves the formulated NLP-problem. This third step happens on the server side and is a black box for the user. Behind the scenes, the server consequently runs several programs organized with the help of RESTful web services. On completion, the solver outputs a set of variables defining the optimal solution which are used to build the desired polyhedron during the fourth step of the algorithm.

The author has tested several solvers for nonlinear programming problems presented on neos-server.org. The tests concerned the problems of finding the optimal cut in convex gemstones. The problems' characteristics were as follows: around 1,000 variables, around 4,000 nonlinear constraints and 3,000 linear constraints. Among the tested solvers were CONOPT, Ipopt, Knitro, LANCELOT, LOQO, MINOS, MOSEK, SNOPT. The objective functions of solutions differed in insignificant figures. Ipopt solver showed the best average time for the given set of problems – 3.50 seconds. The paid solvers MINOS and SNOPT also showed good timings of 5.33 and 4.97 seconds, respectively. Solvers for global optimization have also been tested on simple problems of inscribing convex platonic solids within each other. The Couenne solver showed the best results for these prob-

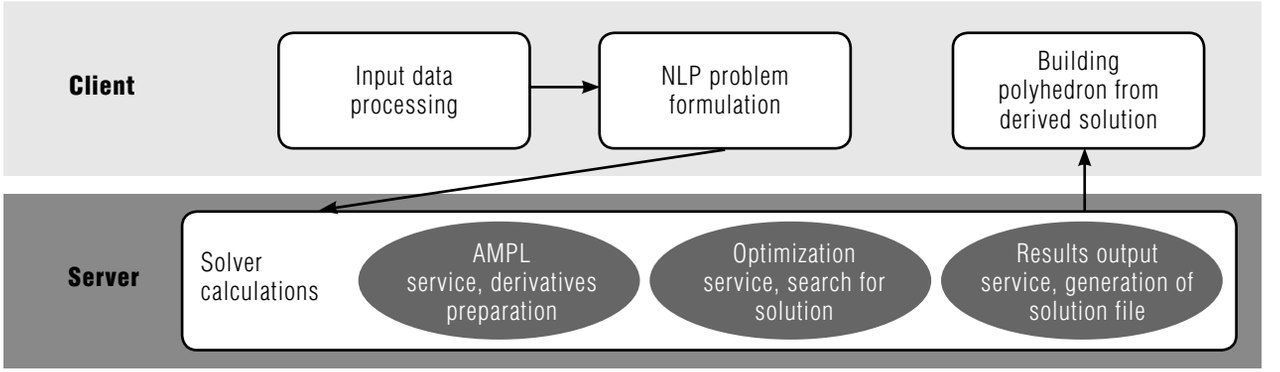

Fig. 1. Client-server application scheme

lems; however, on average, it worked around 300 times longer than Ipopt and this is unacceptable in conditions of production. Moreover, Ipopt was also able to find global maximums for these simple problems.

Accordingly, for the computations of the given NLP-problem the author chose the Ipopt (Internal Point OPTimization) solver [8], as the optimization service. This solver is designed to find a local optimum of a nonlinear programming problem using an interior point method. To perform necessary computations, the server needs callback-functions to calculate the following variables:

- ✦ the values of the objective function $f(x)$;
- ✦ the gradient of the objective function $\nabla f(x)$;
- ✦ the ordinate vector of the constraint vector function $g(x)$;
- ✦ the Jacobian of the constraint vector function $\nabla g(x)^T$;
- ✦ the Hessian of the extended Lagrange function

$$\sigma_f \nabla^2 f(x) + \sum_{i=1}^{m} \lambda_i \nabla^2 g_i(x).$$

If one uses the AMPL programming language [9] for the problem formulation, the language itself provides all these functions for given $f(x)$ and $g(x)$. Thus, one only needs to take care of formulating the NLP-problem in the AMPL format (the formulation of an NLP-problem is discussed in the Chapter 3).

One of the advantages of the client-server approach is the possibility to process any number of problems simultaneously if given the necessary resources. Moreover, the licenses for commercial use of AMPL and Ipopt on a server are cheaper than a large number of licenses for local applications.

2. Local application

The local application (the box version of the program which is to be installed on personal computers) is needed because most of the software clients are located in developing countries where there is no stable server access through the Internet. AMPL is a paid service and is expensive for commercial use in local applications; to overcome that problem, an interface has been developed and implemented in C++ to work with the Ipopt solver directly, without the use of AMPL. The afore-mentioned callback-functions have been written for a narrow set of constraint functions. These functions have the form of cubic polynomials:

$$\sum c_i x_{i1} x_{i2} x_{i3} + \sum c_j x_{j1} x_{j2} + \sum c_k x_k .$$

The sine and cosine functions have also been implemented.

The derivative matrices for such constraints can be written easily, unlike the derivative matrices for the problem in its general form. For the basic algorithm without additional constraints, it is enough to have this representation of the objective function and constraint functions. Moreover, the majority of geometric constraints can be approximated in such form with good enough precision.

The objective function $f(x)$ and constraints $g(x)$ are stored in a special form. During each iteration, Ipopt refers to the callback-functions written in C++ which calculate the necessary derivatives quickly in the current point (*Figure 2*). It is crucial to minimize the algorithmic computational complexity of the callback-functions, since they are called a significant number of times.

Ipopt is rather resource-intensive in terms of CPU time. Thus, the number of problems that can be solved simultaneously on a computer is equal to the number of cores (or twice the number of multithreading cores). If more problems are launched at the same time, their results will be calculated much more slowly. Ipopt can be adjusted so that one task will be solved on several CPUs concurrently, but this gives a tiny boost to performance and does not have much sense in case of a large number of small tasks.

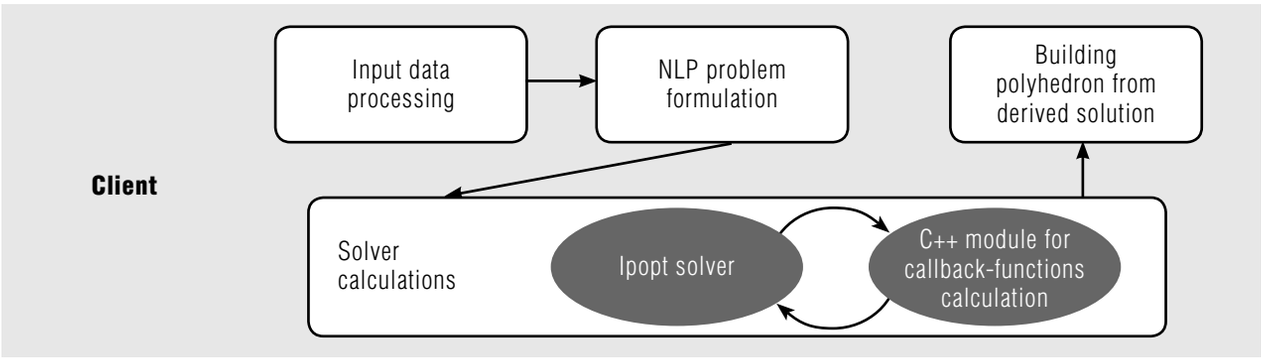

Fig. 2. Local application scheme

For the applied problem being solved in this article, one of the important conditions is to get the solution for any gemstone in limited time, most often in just a minute. In practice, it happens that whatever fixed settings have been chosen, there will always be gems on which Ipopt works longer than the allotted time. That is the reason why all eight runs (keeping in mind a standard modern quad-core processor with multithreading) are performed for the same gemstone with different parameter settings. This approach guarantees getting a result and allows cutting gems of different quality. Gems of different quality are needed because in the jewelry industry the value of a cut gemstone depends not only on its mass. Sometimes it is more profitable to cut a bigger gem of worse quality, and sometimes – a smaller gem of perfect quality.

Apart from that, it turned out that international quality standards are not suitable for appropriate work with nonsymmetrical cut diamonds. A cut diamond, in addition to 3-dimensional symmetry, has a property of optical symmetry, which correlates to how well and symmetrically it sparkles. *Figure 3* shows the course of light in a cut diamond. The more symmetrical the picture, the better the optical symmetry of the cut. The left image presents an optical picture of a perfectly regular cut of 1.0149 carat which was taken as the starting point of the algorithm. The mass of the uncut stone in this example is 1.2904 ct. The right image shows an optical picture of the algorithm result where only the constraints from the international standard of cut diamond quality estimation have been taken into account. The mass of this solution is 1.0601, and it is formally considered to be of highest quality. However, one can easily see that its optical symmetry is far from perfect, therefore the cut diamond will sparkle chaotically and will be difficult to sell. This served as a reason to introduce additional parameters to control optical symmetry of diamonds. The eight simultaneous runs of the algorithm work in such a way that the solutions range from prefect optimal symmetry to the largest mass with acceptable quality. For the reviewed example, the algorithm outputs the following range of masses: 1.0601, 1.0428, 1.0406, 1.0368, 1.0356, 1.0316, 1.0315, 1.0268. All the solutions starting from the mass of 1.0428 ct have beautiful enough optical symmetry, while the solution with the mass of 1.0268 ct hardly differs from perfect optical symmetry. A user-friendly interface has also been developed to allow users to run the algorithm with any desirable settings, customize the program for their specific needs and get the cuts of required quality.

3. Nonlinear programming problem

The original geometric problem needs to be formulated in terms of nonlinear programming.

The volume of the final cut serves as the objective function $f(x)$. It is calculated using the vertices' coordinates through splitting the gem's facets into triangles and presenting the total volume as a sum of simplexes' volumes.

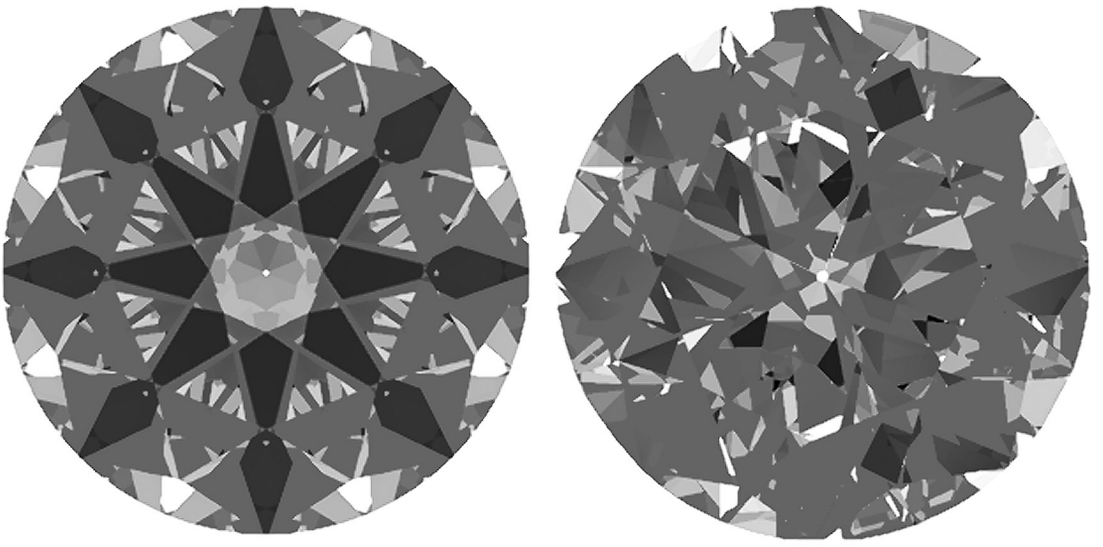

Fig. 3. Optical symmetry of a cut diamond

The variables in this problem fall into two types. The first type comprises the parameters of the cut's facets y_{ap}, y_{bp}, y_{cp}, y_{dp} , where p is the index of a cut's facet. The constraints on the variables of the first type (xU and xL) are set as the initial values of the parameters plus or minus some specific variable, respectively. The value of this variable depends on the extent to which it is allowed to change the angles of the normals to the facets of the given combinatorial structure. The parameters of a gem's facets never change, hence are treated as constants. The second type of variables comprises the parameters responsible for arranging the cut's vertices in space, $x_{ji} \in (-\infty, +\infty)$, where j is the index of a vertex and i refers to a coordinate axis. Since a rather precise algorithm is used to set the starting point, it is possible to speed up the calculation process by setting such constraints on the vertices' coordinates so that the vertices cannot move from the initial position further than some variable dx.

Let us proceed to the description of the general constraints $g(x)$.There are several main groups of constraints for the given problem that are bound to be considered. The first group consists of the constraints to preserve the combinatorial structure of the cut:

$$y_{ap}x_{j1} + y_{bp}x_{j2} + y_{cp}x_{j3} - y_{dp} = 0,$$

where p is a cut's facet index, j is a vertex index, and the indices 1, 2, 3 refer to the three coordinate axes.

This group of constraints is written out for all pairs of vertexes and facets for which it is true that the vertex belongs to the facet. The list of all such pairs can be derived from the original presentation of the polyhedron's facets.

The second group refers to the constraints on the convexity of the required polyhedron. They are written out for each pair of "vertex – facet" where the vertex does not belong to the facet:

$$y_{ap}x_{j1} + y_{bp}x_{j2} + y_{cp}x_{j3} - y_{dp} \le 0.$$

Since the pairwise convexity of all the adjacent facets serves as a sufficient condition for the convexity of the whole polyhedron, these constrains may be written out only for all the pairs of adjacent facets.

The last group of mandatory conditions deals with the fact that the cut has to stay inscribed in the gemstone. These conditions are written as follows:

$$A_r x_{j1} + B_r x_{j2} + C_r x_{j3} - D_r \le 0,$$

where r is the index of a gem's facet, and j is the index of a cut's vertex.

It is worth noting that A_r, B_r, C_r, D_r are constants, not variables. Therefore, this group of constraints is linear and does not have a great influence on the execution time of the solver.

Apart from the mandatory constraints, it is also necessary to outline the constraints on the cut's symmetry. For the case of a convex gemstone (i.e. a partially cut gem) you can read more about NLP-problem formulation in the articles [7, 10]. Let us move on to the nonconvex case.

For the case when the gemstone is nonconvex, additional constraints are necessary which are responsible for separating the cut from the nonconvex parts of the gem. First, a convex shell of the gem is built. For the constraints described above the facets of this convex shell are used. Next, it is necessary to figure out which facets of the gem are nonconvex. This is preformed using the criterion that a facet is nonconvex if there are two vertices in the polyhedron which lie on different sides of the facet. Then, for each nonconvex facet a separating plane is created. The separating plane is defined by the free variables y_a, y_b, y_c, y_d. Constraints are imposed on the separating plane such that all the vertices of the nonconvex plane bound to it lie on one side of the separating plane while all the vertices of the cut lie on the opposite side of it. These equations are assigned to each nonconvex facet of the outer polyhedron. If the maximum deviation from the initial position dx is used for the cut's vertices, then the separating plane may be used to set apart not all the vertices of the inscribed polyhedron, but only the vertices of such facets as may intersect the nonconvex plane of the outer polyhedron. Knowing the value of dx, it is easy to determine the set of such facets for each nonconvex plane of the outer polyhedron. To simplify the work of the solver, another constraint may be added on the length of the normal to a separating plane to be close to 1.0.

Conclusion

The algorithm has been implemented as a client-server application and a local application. It has been tested on problems of different scales. The problems varied from ordinary study examples with simple, similar or easily inscribed polyhedrons with the number of vertices less than fifty to real applied problems with the number of vertices ranging from one hundred to several thousands in the nonconvex case and with the constraints on the symmetry of different severity.

To test the algorithm, a system of remote testing has been written in Python which launches the algorithm on

a given database of gems and builds a report with data about all the runs and all the controlled parameters. The testing system is described in more detail in [7].

In real cases, the algorithm improves results of other algorithms from the same application area by 1.5–5% of the volume depending on the constraints imposed on the symmetry. The average gain in mass of the solutions that may be admitted to cutting is 3%.

Main test examples comprised an inscribed polyhedron with the number of vertices and facets around 150 and an outer polyhedron with the number of vertices and facets around 500; the total number of constraints was around 10,000. For examples with a large number of nonconvexities the number of constraints could reach around 40,000. The time to build the problem for these examples approximates one second. The computation time on the solver without symmetry constraints is around 5 seconds. The average computation time on the solver with standard symmetry constraints is 20 seconds. For nonconvex gems, the average time is 29 seconds. All the measurements have been made using Intel(R) Core(TM) i7-2600 CPU @ 3.40 GHz. To imitate weaker computers, additional tests have been made with reduced CPU frequency. For the maximum frequencies of 0.9, 1.4, 1.9, 2.4, 2.9 GHz, the average computation times for the problem with constraints are 90, 57, 42, 34, 28 seconds, respectively.

The article discusses two implementations of the algorithm: one with the use of a remote computational resource and one without it. The advantages and disadvantages of both implementations are presented. The algorithm itself is described from a mathematical point of view; its application details and special characteristics are also reviewed. As compared to the previous article [7], a method has been found to solve the nonconvex case of the problem which is a significant step forward from the mathematical point of view. It also allows us to apply the algorithm during all the stages of gemstone cutting. The average working time of the algorithm has been estimated, including that on weak processors. At this stage, the algorithm has been implemented for the most common round cut. In future, it is planned to extend it for other types of cuts. ■

References

1. Tarasov A., Musin O. (2010) The strong thirteen spheres problem. *Topology, Geometry, and Dynamics: Rokhlin Memorial*. Saint Petersburg, Russia. 11–16 January 2010.
2. Musin O.R. (2003) Problema dvadtsati pyati sfer [The problem of twenty-five spheres]. *Russian Mathematical Surveys*, vol. 58, no. 4 (352), pp. 153–154 (in Russian).
3. Hales T. (1994) The status of the Kepler conjecture. *Mathematical Intelligencer*, no. 16, pp. 47–58.
4. Valiakhmetova Yu.I., Filippova A.S. (2014) Teoriya optimal'nogo ispol'zovaniya resursov L.V. Kantorovicha v zadachakh raskroya – upakovki: Obzor i istoriya razvitiya metodov resheniya [The Kantorovich's theory of optimal use of resources in the cutting and packing problem]. *Vestnik UGATU*, vol. 18, no. 1 (62), pp. 186–197 (in Russian).
5. Afanas'ev A.P., Voloshinov V.V., Lisov A.A., Naumtseva A.K. (2012) Organizatsiya raspredelennoy obrabotki dannykh s pomoshch'yu RESTful-veb-servisov [Organization of distributed data processing using RESTful-web-services]. *Modern problems of science and education*, no. 6 (Technical sciences), p. 31 (in Russian).
6. Voloshinov V.V., Smirnov S.A. (2012) Printsipy integratsii programmnykh resursov v nauchnykh vychisleniyakh [Principles of program resources integration in scientific calculations]. *Information technologies and computer systems*, no. 3, pp. 66–71 (in Russian).
7. Kokorev D.S. (2016) Optimizatsionnyy algoritm poiska vpisannogo mnogogrannika maksimal'nogo ob''ema [An optimization algorithm of determining an inscribed polyhedron of maximal volume]. *Program products and systems*, no. 1, pp. 90–95 (in Russian).
8. Kawajir Y., Laird C., Wächter A. (2010) *Introduction to Ipopt: A tutorial for downloading, installing, and using Ipopt*. Available at: http://www.coin-or.org/Ipopt/documentation/ (accessed 15 May 2017).
9. Fourer R., Gay D.M., Kernighan B.W. (2003) AMPL: *A modeling language for mathematical programming*. Belmont: Duxburry Press.
10. Kokorev D.S. (2013) Algoritm poiska vypuklogo mnogogrannika maksimal'nogo ob"ema, vpisannogo v drugoy mnogogrannik [An algorithm for search of the convex polyhedrons of maximal volume inscribed in another polyhedrons]. *Information technologies and computer systems*, no. 3, pp. 27–31 (in Russian).

Алгоритм поиска оптимального варианта огранки драгоценного камня максимальной массы с заданными отклонениями от симметричности[2]

Д.С. Кокорев

аспирант лаборатории распределенных вычислительных систем
Институт проблем передачи информации им. А.А. Харкевича РАН
Адрес: 127051, г. Москва, Большой Каретный пер., д. 19, стр. 1
E-mail: korvin-d@yandex.ru

Аннотация

В статье рассматривается задача нахождения многогранников заданной формы внутри других невыпуклых многогранников. Данная задача является частным случаем третьей части 18-й проблемы Гильберта. Она имеет практическое применение в компьютерном моделировании трехмерных объектов, автономном перемещении роботов, ювелирной промышленности. Эта математическая задача интересна с точки зрения ее применения для нахождения огранок драгоценных камней внутри неограненного камня.

В статье предлагается метод поиска вписанных многогранников, основанный на сведении данной задачи к задаче нелинейного программирования и решения ее с помощью готовых программных средств. Основной идеей является то, что задача легко описывается в терминах нелинейного программирования. Целевой функцией является объем искомого многогранника. Ограничения включают в себя сохранение комбинаторной структуры, содержание многогранника внутри другого, выпуклость и дополнительные ограничения, необходимые для практических целей.

В статье рассмотрены две реализации алгоритма: клиент-серверное приложение и локальное приложение. Приведены их достоинства и недостатки. Алгоритм описан не только с математической точки зрения, но и с точки зрения некоторых его прикладных особенностей. По сравнению с предыдущими статьями автора добавлен метод, который позволяет решать невыпуклый случай задачи, что является значительным шагом с математической точки зрения. Кроме того, это позволяет использовать алгоритм на всех этапах огранки драгоценных камней. В конце статьи даны актуальные оценки эффективности и времени работы алгоритма, в том числе на слабых процессорах, и описаны планы дальнейшего развития алгоритма.

Ключевые слова: выпуклый многогранник, комбинаторная структура, вписанный многогранник, огранка драгоценного камня, отклонение от симметричности, задача нелинейного программирования, солвер.

Цитирование: Kokorev D.S. An algorithm for determining the optimal variant of a cut gem with maximal mass and specified symmetry deviations // Business Informatics. 2017. No. 2 (40). P. 40–46. DOI: 10.17323/1998-0663.2017.2.40.46.

Литература

1. Tarasov A., Musin O. The strong thirteen spheres problem // Topology, Geometry, and Dynamics: Rokhlin Memorial. Saint Petersburg, Russia. 11–16 January 2010.
2. Мусин О.Р. Проблема двадцати пяти сфер // Успехи математических наук. 2003. Т. 58, № 4 (352). С. 153–154.
3. Hales T. The status of the Kepler conjecture // Mathematical Intelligencer. 1994. No. 16. P. 47–58.
4. Валиахметова Ю.И., Филиппова А.С. Теория оптимального использования ресурсов Л.В. Канторовича в задачах раскроя – упаковки: Обзор и история развития методов решения // Вестник УГАТУ. 2014. Т. 18, № 1 (62). С. 186–197.
5. Афанасьев А.П., Волошинов В.В., Лисов А.А., Наумцева А.К. Организация распределенной обработки данных с помощью RESTful-веб-сервисов // Современные проблемы науки и образования. 2012. № 6 (приложение «Технические науки»). С. 31.
6. Волошинов В.В., Смирнов С.А. Принципы интеграции программных ресурсов в научных вычислениях // Информационные технологии и вычислительные системы. 2012. № 3. С. 66–71.
7. Кокорев Д.С. Оптимизационный алгоритм поиска вписанного многогранника максимального объема // Программные продукты и системы. 2016. № 1. С. 90–95.
8. Kawajir Y., Laird C., Wächter A. Introduction to Ipopt: A tutorial for downloading, installing, and using Ipopt. [Электронный ресурс]: http://www.coin-or.org/Ipopt/documentation/ (дата обращения 15.05.2017).
9. Fourer R., Gay D.M., Kernighan B.W. AMPL: A modeling language for mathematical programming. Belmont: Duxburry Press, 2003.
10. Кокорев Д.С. Алгоритм поиска выпуклого многогранника максимального объема, вписанного в другой многогранник // Информационные технологии и вычислительные системы. 2013. №3. С. 27–31.

[2] Работа выполнена при поддержке Российского научного фонда (проект 16-11-10352)