

METHODS AND SYSTEMS FOR OPTIMIZATION PROBLEM TRANSFORMATION FOR FACILITATED RESOLUTION

TECHNICAL FIELD

[001] The present invention generally relates to computer-implemented optimization, and more particularly relates to methods and systems for optimization problems transformation for facilitated resolution.

BACKGROUND OF THE DISCLOSURE

[002] Mathematical optimization problems arise in many fields and disciplines, including chemistry, logistics, and machine learning. In many cases, such problems can be mapped onto (i.e., represented as an instance of) Ising spin systems or quadratic unconstrained binary optimization (QUBO) problems.

[003] A possible angle to tackle such optimization problems is to make use of Monte-Carlo algorithms, like the Metropolis-Hastings algorithm or simulated annealing, which allow one to sample from the space of all possible solutions and potentially find or get close to the best configuration(s) of variables. One issue with such methods is that the samples they produce are often correlated due to the locality of transitions considered when exploring the space, giving only the chance to visit a small part of the entire space of solutions.

[004] To circumvent this issue, one can turn to cluster Monte-Carlo algorithms to allow non-local configurations updates and potentially improve exploration. One cluster Monte-Carlo method of interest that has been studied recently is the Houdayer algorithm. This Monte-Carlo move and its derivatives, exhibit the interesting property that they are able to perform isoenergetic moves, i.e., moves that preserve the total energy of the configurations given as input.

[005] Although it is useful in many situations, in practice the Houdayer move and its derivatives have the downside that, to be successful, they require the QUBO formulation of the optimization problem to be such that the number of pairwise interactions between variables is small. Equivalently, the graph underlying the problem should not be too connected.

[006] Thus, there is a need to provide a method and system for transforming an optimization problem in order to facilitate its resolution and provide a higher chance of successful optimization by, for example, enabling the use of the Houdayer move. Furthermore, other desirable features will become apparent from the subsequent detailed description and the appended claims, taken in conjunction with the accompanying drawings and this background of the disclosure.

SUMMARY

[007] According to at least one aspect of the present embodiments, a method for transformation of an optimization problem to facilitate its resolution is provided. The method includes casting the optimization problem into a quadratic unconstrained binary model and transforming the optimization problem into an optimization problem having the same quadratic unconstrained binary model with reduced connectivity. Transforming the optimization problem into the reduced connectivity optimization problem includes partitioning decision variables in the quadratic unconstrained binary model into one or more groups, each of the one or more groups comprising more than one decision variable node and introducing a register variable node between adjacent pairs of the more than one decision variable node in the one or more groups to hold partial values of a sum in a linear constraint to form the optimization problem with reduced connectivity.

[008] According to another aspect of the present embodiments, a system for transformation of an optimization problem to facilitate its resolution is provided. The system includes one or more processors and a storage means comprising instructions for controlling the one or more processors. The instructions include instructions for controlling the one or more processors to cast the optimization problem into a quadratic unconstrained binary model and transform the optimization problem into an optimization problem having the quadratic unconstrained binary model with reduced connectivity. The instructions to transform the optimization problem into an optimization problem with reduced connectivity include instructions to partition decision variables in the quadratic unconstrained binary model into two or more groups, each of the two or more groups comprising at least one decision variable node and instructions to introduce a register variable node between adjacent pairs of the at least one decision variable node in the two or more groups to hold partial values of a sum in a linear constraint to form the optimization problem with reduced connectivity.

BRIEF DESCRIPTION OF THE DRAWINGS

[009] The accompanying figures, where like reference numerals refer to identical or functionally similar elements throughout the separate views and which together with the detailed description below are incorporated in and form part of the specification, serve to illustrate various embodiments and to explain various principles and advantages in accordance with present embodiments.

[0010] FIG. 1 depicts an illustration of a conventional 2 by 3 Ising lattice of spins.

[0011] FIG. 2 depicts a flowchart of reducing connectivity for problem optimization in accordance with the present embodiments.

[0012] FIG. 3 depicts an illustration of introducing register variables between decision variables in accordance with present embodiments.

[0013] FIG. 4 depicts an illustration of a further topology of register variables and decision variables in accordance with the present embodiments.

[0014] FIG. 5 depicts an illustration of an exemplary resulting formulation after applying ReduCon in accordance with the present embodiments.

[0015] FIG. 6 depicts a graph of densities versus Houdayer success rates when using methods for all possible groupings with six variables in accordance with the present embodiments in an example where coefficients in the linear constraint are all equal.

[0016] FIG. 7 depicts a graph of densities versus Houdayer success rates when using methods for all possible groupings with six variables in accordance with the present embodiments in an example where coefficients in the linear constraint are different.

[0017] FIG. 8 depicts an illustration of the Houdayer move in a two-dimensional lattice in accordance with the present embodiments.

[0018] FIG. 9, comprising FIGs. 9A to 9C, depicts an exemplary illustration of a register forming a barrier and allowing creation of more than one connected component in a local overlap in accordance with the present embodiments, wherein FIG. 9A depicts a first configuration, FIG. 9B depicts a second configuration, and FIG. 9C depicts the local overlap of the first and second configurations.

[0019] FIG. 10 depicts an illustration of an exemplary resulting graph after applying ReduCon in accordance with the present embodiments where each group is formed by a single variable.

[0020] FIG. 11 depicts an illustration of an exemplary resulting graph after applying ReduCon in accordance with the present embodiments and the last register is removed.

[0021] FIG. 12 depicts a diagram of steps of a Houdayer move when using methods in accordance with the present embodiments.

[0022] And FIG. 13 depicts a graph of mean relative error together with standard deviation plotted as a function of iteration number for multiple aggregated parallel tempering procedures for different types of Monte-Carlo moves in accordance with the present embodiments.

[0023] Skilled artisans will appreciate that elements in the figures are illustrated for simplicity and clarity and have not necessarily been depicted in full detail.

DETAILED DESCRIPTION

[0024] The following detailed description is merely exemplary in nature and is not intended to limit the invention or the application and uses of the invention. Furthermore, there is no intention to be bound by any theory presented in the preceding background of the invention or the following detailed description. It is the intent of present embodiments to provide systems and methods for transforming optimization problems that can be formulated as Ising models or Quadratic Unconstrained Binary Optimization (QUBO) models into an equivalent formulation that allows a successful use of the Houdayer algorithm. Thus, in accordance with present embodiments, general optimization problems comprised of a quadratic objective function of binary variables together with linear equality constraints involving these variables are considered, as they can be easily cast into an Ising/QUBO formulation by turning linear constraints into quadratic penalties. In particular, the systems and methods in accordance with the present embodiments provide optimization solutions even for problems whose Ising/QUBO formulations have interactions between every pair of variables, the transformation reducing connectivity in the formulation by introducing

new binary variables which allow rewriting quadratic penalties in a different manner, where “connectivity” refers to the structure of interactions in the Ising/QUBO formulation. To achieve this, the variables in the problem are partitioned into multiple groups which allow one to break down the computation of each sum involved in each linear constraint by saving intermediate values of those sums into registers. Thus, new binary variables that act as registers are introduced in order to hold partial values of intermediate sums in a linear constraint. With this augmented set of variables, the original constraints can be rewritten accordingly to make use of the new variables acting as registers, hence leading to a new Ising/QUBO formulation with a reduced connectivity overall. An advantage of considering problems in the form a quadratic objective function with linear constraints is that they encompass a large number of optimization problems such as Traveling Salesman problems, Maximum Cut problems, Number Partitioning problems, Knapsack problems, Minimum Vertex Cover problems and Graph Colouring problems.

[0025] Further, the following definitions, notation and conventions are used. When considering an optimization problem, a “state” is used to refer to any configuration of the underlying variables in a general sense (not necessarily in the context of an optimizer). For instance, a “state space” is referred to independently of any optimization procedure within it. In addition, a “configuration” is any combination of the underlying problem variables that might be visited during exploration of the state space by an optimizer and a “solution” is the final configuration returned by the optimizer, i.e., returned once the optimizer termination criteria have been satisfied. For an optimization problem with n binary variables (which take value in $\{0,1\}$), variables are represented with x_i , where $i = 1, 2, \dots, n$. When collecting all variables into a vector, the resulting object is denoted as $x \in \{0, 1\}^n$. And for an optimization problem with n

spin variables (which take value in $\{-1, 1\}$), variables are represented with σ_i , where $i = 1, 2, \dots, n$. When collecting all variables into a vector, the resulting object is denoted as $\sigma \in \{-1, 1\}^n$.

[0026] The methods and systems in accordance with the present embodiments are particularly applicable to optimization problems which can be mapped onto a constrained optimization of the form

$$\begin{aligned}
 & \text{minimize} && \sum_{i=1}^n \sum_{j=1}^n Q_{ij} x_i x_j \\
 & \text{subject to} && \sum_{i=1}^n b_{1i} x_i = b_1 \\
 & && \vdots \\
 & && \sum_{i=1}^n b_{mi} x_i = b_m \\
 & && x_i \in \{0, 1\} \quad \forall i \in \{1, \dots, n\},
 \end{aligned} \tag{1}$$

namely a quadratic objective function with m linear equality constraints, and n binary variables $x_1, \dots, x_n$. In fact, it is shown hereinafter that advantageously linear inequality constraints can also be taken into account. The variables x_i of the original problem are referred to as “decision variables”. In addition, for simplicity in the discussion of the methods and systems in accordance with the present embodiments hereinafter, the case with $m = 1$ linear constraint is mainly treated, but those skilled artisans will realize that the schemes discussed hereinafter can be generalized to quadratic objective functions with any number m linear constraints. While a technique to handle the general case of real numbers is discussed hereinafter, for simplicity in the general discussion herein, the coefficients b_{ij} and the numbers b_i are assumed to be all positive integers.

[0027] Optimization problems that follow the formulation of Equation (1) can be readily cast into a QUBO formulation. A QUBO (short for “Quadratic Unconstrained

Binary Optimization” as indicated above) problem is an optimization problem that can be formulated as the minimization of a function of the form

$$Q(x) = \sum_{i=1}^n \sum_{j=i}^n Q_{ij} x_i x_j \quad (2)$$

where $x_i \in \{0, 1\} \forall i = 1, \dots, n$. As the name suggests, these optimization problems are only concerned with minimizing a quadratic objective function with binary variables since there are no constraint. To cast a problem formulated as in Equation (1) into a QUBO formulation only requires turning linear equality constraints into quadratic penalties, which is a technique that is used typically in certain approaches to solving optimization problems. Hence, the QUBO formulation corresponding to an optimization problem in the form of Equation (1) amounts to minimizing the function

$$\sum_{i=1}^n \sum_{j=i}^n Q_{ij} x_i x_j + \sum_{j=1}^m \left(\sum_{i=1}^n b_{ji} x_i - b_j \right)^2. \quad (3)$$

[0028] In the QUBO model, the interest is in finding the assignment x^* which minimizes the objective function Q , that is, compute $x^* = \min_{x \in \{0,1\}^n} Q(x)$.

[0029] It should be further noted that a model equivalent to the QUBO one is the Ising model. The Ising model was initially presented as a mathematical model for representing physical systems, such as magnets. The Ising model consists of variables called spins, which can take values -1 or 1 , and which interact with each other according to a given graph structure. Such a graph structure is shown in FIG. 1. FIG. 1 depicts an illustration 100 of a conventional Ising lattice 110 for a 2 by 3 system. The lattice 110 is a 2 by 3 lattice of spins, where first spins 120 having a value -1 are depicted in black and second spins 130 having a value 1 are depicted in white.

[0030] The model assigns a value (often called “energy”) to each configuration of spins through a particular function (often called “Hamiltonian”) of the following form

$$\mathcal{H}(\sigma) = - \sum_{i,j=1}^n J_{ij} \sigma_i \sigma_j - \sum_{i=1}^n h_i \sigma_i \quad (4)$$

where $\sigma_i \in \{-1, 1\} \forall i = 1, \dots, n$. In order to map an Ising formulation to a QUBO one, the range of the variables are changed from $\{-1, 1\}$ to $\{0, 1\}$, which is achieved by substituting each variable σ_i with $2x_i - 1$. Similarly, one can start from a QUBO problem and substitute each variable x_i with $(\sigma_i+1)/2$ to retrieve the corresponding Ising formulation. Thus, hereinafter, a mention of an Ising/QUBO formulation can refer to any optimization problem which demands the minimization of a quadratic objective function with variables that can only take two arbitrary distinct values, and where there is no further constraint on the variables.

[0031] Similar to the QUBO model, in the Ising model, the interest is in finding the assignment which minimizes the function $\mathcal{H}$ in Equation (4), that is, compute $\sigma^* = \min_{\sigma \in \{-1, 1\}^n} \mathcal{H}(\sigma)$. One is often also interested in sampling configurations with low-energy. Note that for either of the two models, adding a constant term to the function to be minimized does not alter the structure of the energy spectrum and, in particular, the lowest energy configurations remain unchanged.

[0032] A convenient way to visualize an Ising/QUBO formulation is through a graph $G = (V, E)$, which is a structure containing a set of objects V (called “nodes”) linked together by a set of “edges” E , where each edge links a pair of nodes. In the context of Ising/QUBO problems, the corresponding graph has the set of nodes $V = \{1, \dots, n\}$, where each node $i \in V$ corresponds to the variable x_i in the problem. Each possible pair of nodes $(i, j) \in V \times V$, which represent the term $Q_{ij}x_ix_j$ that appear in the sum of Equation (2), belongs to the set of edges only if $Q_{ij} \neq 0$. As an example of an Ising formulation viewed as a graph, one can consider an Ising lattice, where spins are placed on a rectangular grid and connected to their nearest neighbours as shown in the lattice illustration 100 of FIG. 1.

[0033] The connectivity of an Ising/QUBO problem is of particular interest since when applying a Houdayer move, the connectivity plays a large role in how successful the move is. Ising/QUBO problems that are very connected, that is where most coefficients Q_{ij} are non-zero, tend to prevent Houdayer moves from yielding an output which differs from the input, rendering the move useless.

[0034] Since mentions of how connected a graph is are recurrent hereinafter, a commonly used measure called “graph density” is introduced. Let $G = (V, E)$ be an undirected graph. The graph density $d(G)$ of G is defined as the ratio of the number of edges $|E|$ to the maximum possible number of edges, that is $d(G) = \frac{2|E|}{|V|(|V|-1)}$.

[0035] In accordance with the methods and systems of the present embodiments, the optimization problems considered are those that can be formulated as those in Equation (1) since they can be further cast into Ising/QUBO problems. In such a case, it is then possible to use tools like the Houdayer move and its variants. Moreover, the optimization problems considered are not restricted to the study of spin systems in physics, meaning that general optimization problems may be considered where any pairwise interaction can happen. In particular, the methods and systems in accordance with the present embodiments work even for optimization problems which have interactions between every pair of variables. In order to do so, the approaches of the methods and systems in accordance with the present embodiments actively modify the input given to the Houdayer move itself, resulting in a higher chance of successful optimization.

[0036] As an example, the number partitioning problem, such as used in logistics for efficient cargo packing and shipping, can be formulated as an Ising model $C = \{c_1, c_2, \dots, c_n\}$ et $\mathbb{R}^+$ be a finite set of positive numbers. When one wishes to find a partition of C into two disjoint subsets A and $C \setminus A$ such that the sum of the elements in

both sets is equal, i.e., $\sum_{a \in A} a = \sum_{b \in C \setminus A} b$, one can fit this into an Ising formulation by defining the Hamiltonian function set out in Equation (4) as seen in Equation (5):

$$\mathcal{H}(\sigma) = \left(\sum_{i=1}^n c_i \sigma_i \right)^2 = \sum_{i,j=1}^n c_i c_j \sigma_i \sigma_j \quad (5)$$

which is of the desired form $-\sum_{i,j} J_{ij} \sigma_i \sigma_j - \sum_i h_i \sigma_i$ of an Ising model, with $J_{ij} = -c_i c_j$ and $h_i = 0$ and $\forall i, j \in \{1, 2, \dots, n\}$.

[0037] More generally, real-life optimization problems, such as logistics and job shop optimized scheduling, improved biological self-assembly via optimized protein folding, and optimized financial portfolio trading, can be expressed in the form of Equation (1), so that the corresponding Ising/QUBO formulation can be used for optimization and reducing connectivity in accordance with the methods and systems of the present embodiments.

[0038] As mentioned hereinabove, the Houdayer move and its derivatives exhibit the interesting property that they are able to perform isoenergetic moves, i.e., moves that preserve the total energy of the input configurations. Although useful in many situations, the Houdayer move has the downside that in practice, to be successful, it requires the number of pairwise interactions between variables to be small. Equivalently, the graph connectivity underlying the problem should not be too connected. To alleviate this issue for an important class of problems and to advantageously provide enhanced solutions for real-world optimization problems, a technique for reducing connectivity is provided in accordance with the methods and systems of the present embodiments where, given a certain Ising/QUBO problem, a new graph structure with reduced connectivity is generated while being equivalent to the original problem. In connection with the methods and systems of the present

embodiments, this technique is referred to as “ReduCon”. ReduCon advantageously provides the skilled person the possibility to apply the Houdayer move to certain very strongly connected real-world problems to provide solutions in situations where other techniques would most likely fail, as the Houdayer move would most likely fail on the original underlying graph.

[0039] Although originally developed for two-dimensional lattices, the Houdayer move has been altered in various works in order to work on lattices in any space dimension. Such variations differ from the ReduCon techniques in accordance with the methods and systems of the present embodiments in several important application-constraining ways. The conventional Houdayer algorithm and its variations tend to yield poor performance in the general case where interactions happen between arbitrary pairs of variables in a system. In addition, application of the Houdayer algorithm and its variations are restricted to certain pre-defined conditions, which are typically known to be favourable for a successful outcome (i.e., the output is different to the input). However, this methodology does not improve the intrinsic success rate of the Houdayer move; it only makes use of the Houdayer move when the input is such that the technique is most likely to work.

[0040] In order to provide optimization solutions even for problems which have interactions between every pair of variables, the methods and systems in accordance with the present embodiments actively modify the input given to the Houdayer move itself, resulting in a higher chance to make it successful.

[0041] In order to make use of the methods and systems in accordance with the present embodiments, one should know how to formulate an optimization problem of interest in the form of Equation (1). As mentioned hereinabove, for simplicity, we

consider in the following such optimization problems but with only one linear equality constraint, yielding the formulation

$$\text{minimize } \sum_{i=1}^n \sum_{j=i+1}^n Q_{ij} x_i x_j \quad (6)$$

$$\begin{aligned} \text{subject to } & \sum_{i=1}^n b_i x_i = b \\ & x_i \in \{0, 1\} \forall i \in \{1, \dots, n\} \end{aligned} \quad (7)$$

[0042] Moreover, the user should be able to represent a solution of the discrete problem through a configuration of the variables x_i 's in the binary formulation specified above. Note that if the problem statement has an inequality constraint $\sum_{i=1}^n b_i x_i \leq b$ instead of an equality, it can be converted to an equality by adding $t = \lfloor \log_2(b) \rfloor + 1$ “slack” variables $s_1, \dots, s_t \in \{0, 1\}$. The constraint then becomes $\sum_{i=1}^n b_i x_i + \sum_{i=1}^t 2^{i-1} s_i = b$, which fits our formulation, but there are now $n + t$ binary variables, namely $x_1, \dots, x_n$ and $s_1, \dots, s_t$. If the problem is specified using Ising variables (that is variables take values in $\{-1, 1\}$) as

$$\text{minimize } \sum_{i=1}^n \sum_{j=i+1}^n J_{ij} \sigma_i \sigma_j + \sum_{i=1}^n h_i \sigma_i \quad (8)$$

$$\begin{aligned} \text{subject to } & \sum_{i=1}^n c_i \sigma_i = c \\ & \sigma_i \in \{-1, 1\} \forall i \in \{1, \dots, n\}. \end{aligned} \quad (9)$$

it can be converted to the formulation in Equation (6) through the substitution $\sigma_i \rightarrow 2x_i - 1$, yielding a new formulation with $x_i \in \{0, 1\} \forall i = 1, \dots, n$ as desired.

[0043] In order to input such a formulation in practice, one needs to specify the values Q_{ij} , b_i and b , which can be given in various ways. For example, the Q_{ij} 's can be specified using an n-by-n matrix Q with elements $(Q)_{ij} = Q_{ij}$, and a list of pairs (i, j) of

indices with the corresponding weight Q_{ij} . The coefficients b_i can be conveniently given as a sequence $[b_1, \dots, b_n]$, while the coefficient b can be given as is.

[0044] To see when the methods and systems in accordance with the present embodiments can be used, we first convert the formulation to a QUBO formulation by turning the linear constraint in Equation (7) into a quadratic penalty, yielding the new objective function

$$Q(x) = \sum_{i=1}^n \sum_{j=1}^n Q_{ij} x_i x_j + \left(\sum_{i=1}^n b_i x_i - b \right)^2 \quad (10)$$

[0045] As mentioned hereinabove, we visualize the QUBO problem as a graph $G = (V, E)$ with vertices $V = \{1, \dots, n\}$, and where there is an edge $(i, j) \in E$, the quadratic term $x_i x_j$ has a non-zero coefficient. The resulting graph visualizing the QUBO problem can thus be more or less connected depending on the number of distinct quadratic terms, and the goal of the ReduCon methods and systems in accordance with the present embodiments is to reduce this connectivity and thereby advantageously improve the success of the Houdayer move and its variants.

[0046] In Equation (10), the first part $\sum_{i=1}^n \sum_{j=1}^n Q_{ij} x_i x_j$ is the initial quadratic objective function that appears in Equation (7) and already induces some edges in the underlying graph. This “initial” connectivity is a constraint in regards to the ReduCon methods and systems of the present embodiments in that it cannot be reduced in accordance with the present embodiments. Since the ReduCon methods and systems in accordance with the present embodiments cannot prevent variables in the “initial” connectivity from being connected, the Houdayer move is bound to perform poorly if the “initial” connectivity is too dense. Accordingly, in accordance with the present embodiments, it is assumed that this first quadratic double-sum $\sum_{i=1}^n \sum_{j=1}^n Q_{ij} x_i x_j$

induces a low number of pairwise interactions, or even none (which happens when $Q_{ij} = 0$ for all $i \neq j$).

[0047] The second part of Equation (10), $(\sum_{i=1}^n b_i x_i - b)^2 = \sum_{i,j=1}^n b_i b_j x_i x_j - 2b \sum_{i=1}^n b_i x_i + b^2$ accounts for the linear constraint and adds edges (i, j) to G whenever $b_i > 0$ and $b_j > 0$. In a worst case, all b_i 's are strictly positive, inducing a fully-connected topology. This is the connectivity that the ReduCon methods and systems in accordance with the present embodiments are able to reduce. And the ReduCon methods and systems in accordance with the present embodiments reduce this connectivity by introducing new variables to the QUBO problem, thereby advantageously enabling application of the Houdayer move and its variants.

[0048] In order to reduce the connectivity, the ReduCon methods and systems in accordance with the present embodiments introduce extra variables which act as intermediate registers to hold partial values of the sum $\sum_{i=1}^n b_i x_i$. This way, not all variables need to interact, since some values are stored into the intermediate registers. In the following, the resulting graph representing the problem is denoted as $G' = (V', E')$.

[0049] Referring to FIG. 2, a flowchart 200 depicts steps of the ReduCon methods in accordance with the present embodiments. Initially, at step 210, an optimization problem following the structure of Equation (6) is cast as an Ising/QUBO formulation for optimization in accordance with the ReduCon methods of the present embodiments.

[0050] At step 220, the set $\{1, \dots, n\}$ representing all decision variables are partitioned into a desired number k of subsets $G_1, \dots, G_k \subseteq \{1, \dots, n\}$. With these subsets, one can define which intermediate values to store. In the present example, the partial sums $S_1 = \sum_{i \in G_1} b_i x_i, S_2 = \sum_{i \in G_1 \cup G_2} b_i x_i$ all the way to $S_k = \sum_{i \in G_1 \cup \dots \cup G_k} b_i x_i$ can be stored. Hereinafter, the subsets $G_1, \dots, G_k$ are sometimes referred to as ReduCon groups.

[0051] At step 230, “register” variables are introduced to hold the values of the intermediate sums (i.e., the partial sums $S_1 = \sum_{i \in G_1} b_i x_i, S_2 = \sum_{i \in G_1 \cup G_2} b_i x_i \dots S_k = \sum_{i \in G_1 \cup \dots \cup G_k} b_i x_i$). Since the sum S_i can take values in $\{0, 1, \dots, \sum_{j \in G_1 \cup \dots \cup G_i} b_j\}$ depending on the values of $x_1, \dots, x_n$, the number of bits necessary to represent the value of the sum is $|R_i| = \lceil \log \sum_{j \in G_1 \cup \dots \cup G_i} b_j \rceil + 1$. Thus, for each S_i a corresponding set with $|R_i|$ binary variables $R_i = \{r_1^{(i)}, \dots, r_{|R_i|}^{(i)}\}$ is introduced which forms a register to hold the value of the sum. In accordance with the methods and systems of the present embodiments, the “register” variables are meant to be binary representations of the actual value of the sum, that is $S_i = \sum_{j=1}^{|R_i|} 2^{j-1} r_j^{(i)}$.

[0052] At step 240, new constraints are introduced so that the new formulation is equivalent to the original one. Indeed, some constraints need to be imposed to ensure that the register variables are indeed holding the values of the intermediate sums, which can be specified through the equalities of Equation (10) to finally rewrite the original constraint $\sum_{i=1}^n b_i x_i = b$ of Equation (7) using the last register’s variables $\sum_{i=1}^{|R_k|} 2^{i-1} r_i^{(k)} = b$:

$$\begin{aligned}
 \sum_{i \in G_1} b_i x_i &= \sum_{i=1}^{|R_1|} 2^{i-1} r_i^{(1)} \\
 \sum_{i=1}^{|R_1|} b 2^{i-1} r_i^{(1)} + \sum_{i \in G_2} b_i x_i &= \sum_{i=1}^{|R_2|} 2^{i-1} r_i^{(2)} \\
 \sum_{i=1}^{|R_2|} 2^{i-1} r_i^{(2)} + \sum_{i \in G_3} b_i x_i &= \sum_{i=1}^{|R_3|} 2^{i-1} r_i^{(3)} \\
 &\vdots \\
 \sum_{i=1}^{|R_{k-1}|} 2^{i-1} r_i^{(k-1)} + \sum_{i \in G_k} b_i x_i &= \sum_{i=1}^{|R_k|} 2^{i-1} r_i^{(k)}
 \end{aligned} \tag{11}$$

[0053] Then, at step 250, the new formulation is converted to a QUBO formulation. By turning constraints into quadratic penalties this yields the new function to minimize as shown in Equation (12):

$$\begin{aligned}
 Q(x) = & \sum_{i=1}^n \sum_{j=1}^n Q_{ij} x_i x_j + \left(\sum_{i \in G_1} b_i x_i - \sum_{i=1}^{|R_1|} 2^{i-1} r_i^{(1)} \right)^2 \\
 & + \left(\sum_{i=1}^{|R_1|} 2^{i-1} r_i^{(1)} + \sum_{i \in G_2} b_i x_i - \sum_{i=1}^{|R_2|} 2^{i-1} r_i^{(2)} \right)^2 \\
 & \vdots \\
 & + \left(\sum_{i=1}^{|R_{k-2}|} 2^{i-1} r_i^{(k-1)} + \sum_{i \in G_k} b_i x_i - \sum_{i=1}^{|R_k|} 2^{i-1} r_i^{(k)} \right)^2 \\
 & + \left(\sum_{i=1}^{|R_k|} 2^{i-1} r_i^{(k)} - b \right)^2
 \end{aligned} \tag{12}$$

inducing a graph $G' = (V', E')$ which, advantageously, is not fully connected—i.e., thereby reducing connectivity in accordance with the ReduCon methods and systems of the present embodiments.

[0054] The resulting connectivity can be determined as follows. As discussed hereinabove, it is assumed that $\sum_{i=1}^n \sum_{j=i}^n Q_{ij} x_i x_j$ induces a low number of interactions between the decision variables. One can show that the new interactions through Equation (12) allow one to lower the connectivity overall. Indeed, the first quadratic penalty $\left(\sum_{i \in G_1} b_i x_i - \sum_{i=1}^{|R_1|} 2^{i-1} r_i^{(1)} \right)^2$, for example, involves variables $\{x_i : i \in G_1\}$ and $\{r_i^{(1)} : i = 1, \dots, |R_1|\}$, the second quadratic penalty $\left(\sum_{i=1}^{|R_1|} 2^{i-1} r_i^{(1)} + \sum_{i \in G_2} b_i x_i - \sum_{i=1}^{|R_2|} 2^{i-1} r_i^{(2)} \right)^2$ involves variables $\{x_i : i \in G_2\}$, $\{r_i^{(1)} : i = 1, \dots, |R_1|\}$ and $\{r_i^{(2)} : i = 1, \dots, |R_2|\}$, and so on. Thus, it is clear that, advantageously, at no point variables coming from non-adjacent registers interact. These missing interactions imply that the resulting graph is not fully-connected, i.e., connectivity can be reduced in accordance with the ReduCon methods and systems in

accordance with the present embodiments. Going through each penalty reveals the structure of the new graph. Referring to FIG. 3, an illustration 300 visualizes the graph structure resulting from the ReduCon methods and systems. In this depiction, it is assumed that variables in different ReduCon groups do not interact. When applying the ReduCon methods and systems, decision variables 310 are grouped into k ReduCon groups $G_1, \dots, G_k$, and register variables 320 are added, forming registers $R_1, \dots, R_k$. Within each circle box 310a to 310e representing ReduCon groups and each circle box 320a to 320d representing registers, variables are fully connected. There are also edges 330 between each decision variable of each ReduCon group and the register variables from previous and next registers (links 330a to 330i). Finally, edges 340 are also added between variables of adjacent registers (links 340a to 340d) in accordance with present embodiments. In the illustration 300, each edge 330 and each edge 340, actually represent interactions between every variable contained within each of the two entities connected by the edge.

[0055] There are multiple ways in which ReduCon can be modified to handle more general constraints in accordance with the methods and systems of the present embodiments. For example, one can consider a problem of the general form of Equation (1) with m constraints. In such a case, after grouping, one needs registers to store partial sums corresponding to each constraint. That is, after choosing k ReduCon groups $G_1, \dots, G_k$ to partition the decision variables into, the l th partial sum for the j th constraint is now $S_l^{(j)} = \sum_{i \in G_1 \cup \dots \cup G_j} b_{ji} x_i$, meaning a corresponding register $R_l^{(j)}$ for $l = 1, \dots, k$ and $j = 1, \dots, m$ must be introduced. Finally, by adding extra constraints to make the problem equivalent, a topology as depicted in the illustration 400 of FIG. 4 results. In the illustration 400 (note not all items are labeled with a reference numeral for simplicity), it is assumed that variables in different ReduCon groups do not interact.

As shown in the illustration 400, when applying the general connectivity reduction process in accordance with the methods and systems of the present embodiments, decision variables 410 are grouped into k groups $G_1, \dots, G_k$ and register variables 420 are added for all constraints, forming registers $R_l^{(j)}$ for $l = 1, \dots, k$ and $j = 1, \dots, m$. Within each circle box representing groups 410 and registers 420, variables are fully connected. There are also edges 430 between each variable of each group and the variables from previous and next registers of the same constraint. In accordance with the present embodiments, edges 440 are also added between variables of adjacent registers corresponding to the same constraint. In the illustration 400, each edge 430 and each edge 440 actually stands for interactions between every variable contained within each of the two entities connected by the edge.

[0056] Another way in which ReduCon can be modified to handle more general constraints in accordance with the methods and systems of the present embodiments is where one considers that the coefficients b_{ij} in the linear constraint(s) are not necessarily positive integers, but real numbers instead. In this case, one needs to change the representation of the registers' values since, in conventional binary representation, they can only handle positive integer values. The simplest way to do so is to use a floating-point representation for each register's value, such as that described in the IEEE 754 standard. Depending on what precision is needed, each register will need a certain number of bits defined by the format chosen. These variables could be binary32 (i.e., 32 binary variables), binary64 (i.e., 64 binary variables), or binary128 (i.e., 128 binary variables).

[0057] In the formulation resulting from the ReduCon methods and systems in accordance with the present embodiments (i.e., the formulation formed at step 250 of the flowchart 200 (FIG. 2)), the variables $x_1, \dots, x_n$ are present as before, but extra

binary variables are introduced for each introduced register. In total, the new number of variables (or nodes $|V'|$ in G') is described by Equation (13):

$$\begin{aligned}
 |V'| &= n + \sum_{i=1}^k |R_i| \\
 &= n + \sum_{i=1}^k \left(\left\lfloor \log \sum_{j \in G_1 \cup \dots \cup G_k} b_j \right\rfloor + 1 \right) \\
 &= n + k + \sum_{i=1}^k \left\lfloor \log \sum_{j \in G_1 \cup \dots \cup G_k} b_j \right\rfloor
 \end{aligned} \tag{13}$$

[0058] The number of edges can be computed by looking at the illustration 300 of FIG. 3 as follows:

$$\begin{aligned}
 |E'| &= \#(\text{edges within each group}) \\
 &\quad + \#(\text{edges within each register}) \\
 &\quad + \#(\text{edges between each group and adjacent registers}) \\
 &\quad + \#(\text{edges between adjacent registers}) \\
 &= \sum_{j=1}^k \binom{|G_j|}{2} + \sum_{j=1}^k \binom{|R_j|}{2} + \sum_{j=1}^{k-1} (|G_j| \cdot |R_j| + |G_{j+1}| \cdot |R_j|) + |G_k| \cdot |R_k| + \sum_{j=1}^{k-1} |R_j| \cdot |R_{j+1}| \\
 &= \sum_{j=1}^k \left(\binom{|G_j|}{2} + \binom{|R_j|}{2} \right) + \sum_{j=1}^{k-1} (|G_j| \cdot |R_j| + |G_{j+1}| \cdot |R_j| + |R_j| \cdot |R_{j+1}|) + |G_k| \cdot |R_k|
 \end{aligned} \tag{14}$$

[0059] As an example, consider the following formulation with five variables:

$$\begin{aligned}
 &\text{minimize} \quad \sum_{i=1}^5 c_i x_i \\
 &\text{subject to} \quad \sum_{i=1}^5 x_i = 1 \\
 &\quad \quad \quad x_i \in \{0, 1\} \quad \forall i \in \{1, \dots, 5\}
 \end{aligned} \tag{15}$$

This example indeed fits the framework of the ReduCon methods and systems, with

$$Q_{ij} = \begin{cases} \sqrt{c_i c_j}, & \text{if } i = j \\ 0, & \text{otherwise} \end{cases}.$$

Once it is converted into QUBO form, the function to optimize

becomes $Q(x) = \sum_{i=1}^5 c_i x_i + \left(\sum_{i=1}^5 x_i - 1 \right)^2$, giving rise to a fully connected topology due to the penalty term (second term), since all variables are part of it. The connectivity

thus has a density $d(G) = 1$. Since the interactions arise from the linear constraint, the ReduCon prescription can thus be used and, for example, create $k = 3$ groups as follows: $G_1 = \{1,2\}$, $G_2 = \{3,4\}$ and $G_3 = \{5\}$ as shown in the illustration 500 in FIG. 5 which depicts the resulting graph after applying ReduCon on the example in Equation (15). In this case, $|R_1| = \lfloor \log(\sum_{i \in G_1} 1) \rfloor + 1 = \lfloor \log(2) \rfloor + 1 = 2$ bits must be introduced for the first register, $|R_2| = \lfloor \log(\sum_{i \in G_1 \cup G_2} 1) \rfloor + 1 = \lfloor \log(4) \rfloor + 1 = 3$ bits must be introduced for the second one, and $|R_3| = \lfloor \log(\sum_{i \in G_1 \cup G_2 \cup G_3} 1) \rfloor + 1 = \lfloor \log(5) \rfloor + 1 = 3$ bits must be introduced for the third one. The resulting graph depicted in the illustration 500 again shows how connectivity is reduced in accordance with the ReduCon methods and systems of the present embodiments.

[0060] The new number of variables is now $V' = 5+2+3+3 = 13$, and so the maximum number of edges is $\binom{V'}{2} = \binom{13}{2} = 78$. Computing the number of edges using Equation (14) as follows:

$$\begin{aligned}
 |E'| &= \sum_{j=1}^k \left(\binom{|G_j|}{2} + \binom{|R_j|}{2} \right) + \sum_{j=1}^{k-1} (|G_j| \cdot |R_j| + |G_{j+1}| \cdot |R_j| + |R_j| \cdot |R_{j+1}|) + |G_k| \cdot |R_k| \\
 &= \binom{2}{2} + \binom{2}{2} + \binom{1}{2} + \binom{2}{2} + \binom{3}{2} + \binom{3}{2} + 2 \cdot 2 + 2 \cdot 2 + 2 \cdot 3 + 3 \cdot 1 + 1 \cdot 3 + 2 \cdot 3 + 3 \cdot 3 \\
 &= 44,
 \end{aligned} \tag{16}$$

yields a graph density of $d(G') = 44/78 \approx 0.564$ for the new graph. Hence, thanks to the transformation induced by the ReduCon methods and systems in accordance with the present embodiments, the original fully-connected problem is turned into a much sparser one.

[0061] Hereinabove, it has been discussed how ReduCon needs to group decision variables into different sets (i.e., into the ReduCon groups) in order to build and store partial values of the sums involved in the linear constraints. The question remains whether there exists an optimal way to do so and, in this context, what optimal means.

Ultimately, the goal of the ReduCon methods and systems in accordance with the present embodiments is to reduce graph connectivity to allow the use of Houdayer moves. This means that it is preferred that for most pairs of configurations, applying the Houdayer move results in a successful transition, one where the resulting pair of configurations is different from the starting one. Hence, what is defined as the optimal groupings of variables is one for which the resulting graph yields the highest chance of success for the Houdayer move, regardless of the pair of configurations given.

[0062] Generally, as discussed hereinabove, the Houdayer move tends to be more successful when the problem at hand has a weakly-connected underlying graph. To obtain a new graph in accordance with present embodiments, the ReduCon methods and systems perform a very specific choice of addition of nodes and how they relate to each other. This means that even though the graph density may be reduced, it could be that certain choices of groupings can only yield unsuccessful Houdayer moves.

[0063] This can be made explicit through simple experiments where, for a certain problem, how different groupings lead to different densities and different rates of success for the Houdayer move are compared. Experiments are performed as follows. For a desired choice of ReduCon groups, apply the ReduCon method in accordance with the present embodiments to an input problem with an underlying graph G to get a new formulation with graph G' , from which a new graph density $d(G')$ can be deduced. Secondly, repeat m times the following three steps (or, if the number of variables n in the problem is low, consider all 2^{2n} possible pairs): (a) sample uniformly at random a pair of configurations $(\sigma^{(1)}, \sigma^{(2)})$ in the original problem; (b) convert the pair of configurations to a corresponding pair $(\sigma'^{(1)}, \sigma'^{(2)})$ in the new formulation G' , and then try to perform a Houdayer move and record whether the Houdayer move is a success

or not. And, thirdly, return the density $d(G')$ together with the “success rate”

$$\frac{1}{n} \sum_{i=1}^m \mathbb{I} \{\text{sample } i \text{ was successful}\}.$$

[0064] Repeating this process for groupings of interest (potentially all groupings), one can collect all resulting densities and success rates and plot them on a two-dimensional plane to visualize the performance of different groupings. For example, assume a problem with six variables $x_1, \dots, x_6$ and linear constraint $\sum_{i=1}^6 x_i = 1$, that is $x_i = 1, \forall i \in \{1, \dots, 6\}$. The process explained above is repeated for all possible groupings and, since the number of variables is low, all $2^{2 \cdot 6} = 4096$ pairs of configurations are considered. A graph 600 in FIG. 6 plots densities versus Houdayer success rates when using methods for all possible groupings with six variables in accordance with the present embodiments when coefficients in the linear constraint are all equal, where each point corresponds to different grouping. Although the trend suggests that a lower density implies more successful Houdayer moves, graphs with different densities can yield the same success rate.

[0065] Suppose instead of having coefficients being equal in the linear equality constraint, coefficients of different integers, such as $b_1 = b_2 = 3, b_3 = b_4 = 2, b_5 = 1, b_6 = 8$, are used. FIG. 7 depicts a graph 700 of the densities versus the Houdayer success rates when using ReduCon methods and systems in accordance with the present embodiments for all possible groupings with six variables in accordance with the present embodiments when coefficients in the linear constraint are different. The results in the graph 700 show that even when a graph’s density is rather large, the success of Houdayer moves can be still appreciably high (note, the point with density ≈ 0.8 and success ratio ≈ 0.12), whereas some lower density graphs might reach lower success rates (e.g., the point with density ≈ 0.65 and success ratio ≈ 0.04). It can thus be concluded that graph density is not the most deciding factor.

[0066] To understand which grouping of variables is best, one should go back to the definition of the Houdayer move. The Houdayer move works with configurations of spins, that is when variables take values in $\{-1,1\}$. To use the Houdayer move with binary variables, one can use the conversion to the formulation in Equation (6) through the one-to-one transformation $\sigma_i = 2x_i - 1$, yielding a new formulation with $x_i \in \{0,1\} \forall i \in 1, \dots, n$. At the heart of the Houdayer move is the selection of multiple variables to be flipped, which is done by first computing the “local overlap” at each site between a pair of spin configurations $(\sigma^{(1)}, \sigma^{(2)})$, which informs about the similarity of spins between the configurations. Formally, the local overlap at site $i \in V$ between the two configurations is defined as $q_i = \sigma_i^{(1)} \sigma_i^{(2)}$. After that, the Houdayer moves picks a group of variables which differ between the two configurations and which form a connected component. Finally, the values of the chosen variables are flipped in both configurations, possibly yielding a new pair of configurations. The process is visualized in illustration 800 in FIG. 8 for a 2-dimensional lattice graph. The illustration 600 depicts a visual representation of the Houdayer move, where spins 805 with value 1 are white, and spins 810 with value -1 are black. The Houdayer move starts with a pair of independent spin configurations $\sigma^{(1)}$ 820 and $\sigma^{(2)}$ 825 and first computes their local overlap 830 $q_i = \sigma_i^{(1)} \sigma_i^{(2)}$ at each site (or vertex) i , revealing where spins differ between the two configurations. One of the sites 835 with overlap value $q_i = -1$ is then chosen uniformly at random and a corresponding cluster 840 is grown from the site 835. After that, the spins in the cluster 840a, 840b are swapped (i.e., flipped) between the two configurations, yielding a new pair $\tau^{(1)}$ 850 and $\tau^{(2)}$ 855.

[0067] If the number of connected components is equal to one, then the Houdayer move has no choice but to pick that component, resulting in a failure in the sense that the resulting pair is identical to the original one, where just the configurations are

swapped. Otherwise, if there is more than one connected component, new pairs of configurations at the same total energy can be reached. Thus, the goal is to have more than one connected component in the local overlap for any pair of configurations.

[0068] To create more than one connected component in the local overlap 840 of two given configurations, the variables which are equal (the nodes 805 in the illustration 800) must form some kind of “barrier” to allow the connected components formed by the nodes 810 to not join together, hence forming multiple clusters. To create such “barriers” in the graph that results from the ReduCon methods and systems in accordance with the present embodiments, we look again at the example discussed in connection with Equation (15). Let $(\sigma^{(1)}, \sigma^{(2)})$ be a pair of configurations in the original problem with underlying graph G , and suppose that the two matching configurations in the new graph structure G' created by the ReduCon methods and systems in accordance with the present embodiments are $(\sigma'^{(1)}, \sigma'^{(2)})$. Referring to FIGs. 9A, 9B and 9C, two such configurations are shown (a first configuration 900 in FIG. 9A and a second configuration 930 in FIG. 9B) together with their resulting local overlap 960 in FIG. 9C. One can observe that even though variables are different in group G_1 and in group G_2 from the first configuration 900 to the second configuration 930, register R_1 has the same value in both configurations 900, 930. Because of this same value in both configurations 900, 930, R_1 forms a barrier 965 creating two connected components in the local overlap 960, the barrier 965 making the variables in G_1 and G_2 distinct clusters and enabling the use of a successful Houdayer move. If variables had differed only in G_1 , then only one cluster would have been observed. In general, one of two things can happen: (a) if variables that are different between the configurations in the pair all belong to the same ReduCon group, they form only one connected component since variables in a group are fully-connected to each other or (b) some variables are in

distinct groups, in which case they can form distinct connected components only if there is no way to connect them, which happens if a register in-between (e.g. R_1) acts as a barrier in the local overlap such as the barrier 965 in the local overlap 960.

[0069] Knowing this, it can be deduced that a good grouping is one that for most pairs of configurations $(\sigma^{(1)}, \sigma^{(2)})$, it is very likely that when converted to a pair $(\sigma'^{(1)}, \sigma'^{(2)})$ in the new formulation with graph G' , at least one of the registers is going to have the same value for both configurations in the pair. To have more opportunities to create “register barriers”, a grouping which yields the greatest number of registers, which is the grouping formed by putting each decision variables x_i in its own ReduCon group, should be used. An example of such grouping is shown in an illustration 1000 in FIG. 10. The illustration 1000 depicts an exemplary resulting graph after applying the ReduCon methods and systems in accordance with the present embodiments. Each group is formed by a single variable and $G_i = \{i\}, \forall i \in \{1, \dots, n\}$. The final decision to make is the order of variables, since even if each group contains one variable, one should choose which group contains which variable. One heuristic which works particularly well is to look in the linear constraint at the coefficients b_i corresponding to variables x_i . If one orders ReduCon groups in such a way that variables which have a same coefficient the greatest number of times appear first, then ReduCon empirically works better. The first variables should be those whose coefficient's value appear most (but their order does not matter), and whenever there are ties, variables should be ordered in increasing order with respect to their coefficients. For example, if coefficients are given by $b_1=73$ $b_2=32$, $b_3=12$ $b_4 = 45$, $b_5 = 45$, $b_6 = 32$, the coefficient value which appears most are 32 and 45. Since both appear twice, there is a tie and so it is solved by taking the smallest coefficient first, that is 32. Hence, the first variables appearing should be those with that coefficient (that is, x_2 and x_6 should be in the first

ReduCon groups) but their order does not matter. Then variables with coefficient 45 should appear (that is, x_4 and x_5), with again their order not mattering. Finally, the coefficients 12 and 73 both only appear once. This tie is resolved by making the variable x_3 appear first (since 12 is the smallest of the two coefficients), followed by x_1 . Hence, an empirically good way to create ReduCon groups could be $G_1 = \{2\}$, $G_2 = \{6\}$, $G_3 = \{4\}$, $G_4 = \{5\}$, $G_5 = \{3\}$, $G_6 = \{1\}$.

[0070] As has been described hereinabove, the creation of “register barriers” is what makes the Houdayer move successful. However, it is clear that the last register cannot act as a barrier since no group comes after it. Thus, the ReduCon methods and systems in accordance with the present embodiments can be slightly changed to simply drop the last register R_k and adjust the constraints depending on the last register accordingly. Specifically, instead of using the two constraints $\sum_{i=1}^{|R_{k-1}|} 2^{i-1} r_i^{(k-1)} + \sum_{i \in G_k} b_i x_i = \sum_{i=1}^{|R_k|} 2^{i-1} r_i^{(k)}$ and $\sum_{i=1}^{|R_k|} 2^{i-1} r_i^{(k)} = b$, one just needs a single constraint $\sum_{i=1}^{|R_{k-1}|} 2^{i-1} r_i^{(k-1)} + \sum_{i \in G_k} b_i x_i = b$. This reduces the number of nodes and edges in the resulting graph G' , which advantageously speeds up the process and yields the same results as the original technique in accordance with the ReduCon methods and systems. Referring to FIG. 11, the result from the ReduCon methods and systems in accordance with the present embodiments when removing the last register nodes can be seen in an illustration 1100 showing how groups and registers are connected in the last-register-removal case. In the illustration 1100, it is assumed variables in different ReduCon groups do not interact. Within each circle box 1110 representing groups and each circle box 1120 representing registers, variables are fully connected. There are also edges 1130 between each decision variable of each group and the register variables from previous and next registers and edges 1140 added between variables of adjacent registers. In the illustration 1100, each edge 1130 and each edge 1140, stands for

interactions between every variable contained within each of the two entities connected by the edge.

[0071] Even with the advantages provided by the ReduCon methods and systems in accordance with the present embodiments, time and storage space required to perform computations in the graph resulting from the ReduCon methods and systems, even though many nodes have been added, is not seriously impacted. The computation of connected components in the local overlap during the Houdayer move can be greatly simplified when the original objective function $\sum_{i=1}^n \sum_{j=1}^n Q_{ij} x_i x_j$ induces no interaction between variables in different ReduCon groups (i.e., when $Q_{ij} = 0$ for two variables x_i and x_j that are in distinct ReduCon groups). In fact, since one already knows the structure of the resulting graph, the connected components can be determined once the registers that act as barriers are located. Hence, no graph structure needs to be stored in this case, reducing the storage space as well the running time of the Houdayer move.

[0072] Herein, a few applications of the ReduCon methods and systems in accordance with the present embodiments which show how they can be used together with the Houdayer move are presented. In all instances considered, the use of a Houdayer move on a pair of configurations $(\sigma^{(1)}, \sigma^{(2)})$ is replaced by multiple steps. Denoting the underlying graph of the original optimization problem by G , the ReduCon methods and systems in accordance with the present embodiments allows one to retrieve the new graph G' and to easily convert configurations between the two formulations.

[0073] Referring to FIG. 12, a diagram 1200 illustrates application of a Houdayer move while using the ReduCon techniques (hereafter referred to as a “ReduCon Houdayer move”) in accordance with the methods and systems of the present embodiments. First, a ReduCon conversion step 1210 converts the pair of

configurations $(\sigma^{(1)}, \sigma^{(2)})$ 1205 from the optimization problem with G to the corresponding pair of configurations $(\sigma'^{(1)}, \sigma'^{(2)})$ 1215 in the optimization problem with G' . Then, a Houdayer move 1220 is performed on G' , yielding a new pair of configurations $(\tau'^{(1)}, \tau'^{(2)})$ 1225 in the optimization problem with G' . Thereafter, the pair of configurations $(\tau'^{(1)}, \tau'^{(2)})$ 1225 from the optimization problem with G' are converted 1230 to a corresponding pair of configurations $(\tau^{(1)}, \tau^{(2)})$ 1235 in the optimization problem with G . G and G' refer to the underlying graphs of the original formulation and the one created by ReduCon, respectively.

[0074] The first obvious application of the ReduCon Houdayer move in accordance with the present embodiments is in the context of Monte-Carlo methods, in particular in the context of Metropolis-Hastings algorithms and their derivatives such as simulated annealing or parallel tempering. These probabilistic algorithms aim to find global optimal solutions of an optimization problem by searching a large state space, and sometimes are content with sampling low-energy configurations.

[0075] The use of Houdayer moves combined with parallel tempering has been shown to speed-up convergence to minima, as it acts as a form of a tunnelling effect by helping to explore solutions with similar levels of energy as the current ones. As indicated hereinabove, the success of the Houdayer move is limited by the connectivity of the graph at hand. This limitation can advantageously be alleviated by performing a ReduCon Houdayer move when the graph becomes too connected. Problems such as Knapsack or MAX 2-SAT can benefit from the ReduCon Houdayer move, making it much easier to rapidly reach low-energy solutions when the ReduCon Houdayer move is combined with usual Monte-Carlo methods. Note also that ReduCon can be used in combination with other versions of the Houdayer move such as the generalization

presented in “The Houdayer Algorithm: Overview, Extensions, and Applications” (<https://arxiv.org/pdf/2211.11556.pdf>).

[0076] Another class of methods which the ReduCon Houdayer move can enhance is one termed “hybrid”. For such methods, the main Monte-Carlo procedure is performed on a non-binary formulation of the optimization problem, but the (ReduCon) Houdayer move is performed on the equivalent Ising/QUBO formulation in accordance with methods and systems of the present embodiments. There is thus a conversion taking place to go from the non-binary formulation to the binary one. This is done because it is sometimes much easier to perform an optimization algorithm (like simulated annealing or parallel tempering) on a formulation of the problem where variables can take up many different values rather than converting it to a complex Ising/QUBO formulation before performing the optimization in the Ising/QUBO formulation. The Ising/QUBO formulation is only used to enable the application of the (ReduCon) Houdayer move, giving once again the opportunity to find pairs of configurations at the same energy level.

[0077] An optimization procedure (like simulated annealing) in this context starts from some pair of configurations of the variables in the non-binary formulation. One step of the optimization algorithm is performed in which a new pair of configurations is found. The pair of configurations is then converted to a corresponding one in the QUBO formulation. The Houdayer move (or the ReduCon Houdayer move if the underlying graph is too connected) is next performed to obtain a new pair of configurations at the same total energy level. The pair of configurations is then converted back to the corresponding pair in the non-binary formulation. Finally, the procedure goes back to performing a step of the optimization algorithm and this process is repeated unless some stopping criterion is met.

[0078] Another application where the methods and systems in accordance with the present embodiments provides beneficial efficiencies as discussed in “The Houdayer Algorithm: Overview, Extensions, and Applications” (<https://arxiv.org/pdf/2211.11556.pdf>) is improving sampling of low-energy configurations. It is known that use of the Houdayer improves the sampling of low-energy configurations, especially where ground states are the ultimate aim. Some conventional strategies make use of the principle that, since the Houdayer move preserves the total energy of the pair of configurations, it must be either that one of the two configurations decreases in energy or that both stay at the same energy level. In general, for this application the methods and systems in accordance with the present embodiments would start with generating some low-energy configurations through typical heuristic methods like simulated annealing to form a set P. The configurations in P are then paired, allowing one to use Houdayer moves to potentially generate more pairs at the same total energy level. If the Houdayer moves are successful, new configurations with lower energy levels are reached and these new configurations are stored in the set P with the previous configurations. As the set P has been augmented with the new configurations with lower energy levels, the configurations in P are paired and more pairs are generated using the Houdayer move unless some stopping criterion is met. The stopping criteria may include a predetermined number of iterations or a determination that no new pairs are being generated at the same total energy level. These steps can easily be customized and improved aimed at focusing the generation towards configurations with lower energy, yielding for example genetic algorithms when a selection process is added.

[0079] More generally, any algorithm that makes use of the Houdayer move benefits from the ReduCon methods and systems in accordance with the present embodiments

as the ReduCon methods and systems offer access to more successful Houdayer moves by reducing connectivity of the original formulation. Additional examples of uses of the Houdayer move which can benefit from the ReduCon methods and systems in accordance with the present embodiments include genetic algorithms making use of the Houdayer during the reproduction step and multi-objective optimization such as those where one is interested in finding Pareto optimal solutions, or more generally, the Pareto front (i.e., the set of Pareto optimal outcomes). In regards to multi-objective optimization interested in finding Pareto optimal solutions, if all objectives are themselves in Ising/QUBO, one can combine them into a bigger Ising/QUBO problem by using a technique called linear scalarization. For example, if there are m objective functions $Q_1(x), \dots, Q_m(x)$ to be optimized, one can consider the new unique objective function $Q(x) = \sum_{i=1}^m w_i Q_i(x)$, where $w_i > 0 \forall i \in \{1, \dots, m\}$. This can then be solved with the help of Houdayer moves, and thus ReduCon Houdayer moves when beneficial. In particular, it can be shown that the solutions to the linear scalarization problem $Q(x)$ is on the Pareto front, no matter the choice of weights. This means that if one is able to get a few solutions to the linear scalarization problem, one could use them together with the ReduCon Houdayer move in accordance with the methods and systems of the present embodiments to potentially generate many more solutions at the Pareto front.

[0080] Those skilled in the art will realize that there are a number of practical applications which can benefit from the ReduCon methods and systems in accordance with the present embodiments. For example, applications in logistics and scheduling can benefit from the ReduCon methods and systems in accordance with the present embodiments. These include logistic planning for cargo shipping, car manufacturing, job shop scheduling or traffic flow optimization. As a shipping example, packages are usually shipped through cargo trucks or planes. While every package has a certain

weight and dimension, a cargo vehicle has a fixed capacity and weight limitation. The cost to the shipper of sending a package usually does not depend on the weight, since dispatchers get paid on a contract. The dispatchers, therefore, try to maximize their profit by packing efficiently and shipping the maximum weight in a fixed volume. Improved solutions to this profit optimization problem can advantageously be obtained by the ReduCon methods and systems in accordance with the present embodiments. Another logistics application for the ReduCon methods and systems in accordance with the present embodiments is optimizing solutions for vehicle routing. The vehicle routing problem is an important combinatorial optimization problem in which the goal is to find an optimal set of routes for a fleet of vehicles that deliver goods from an origin (depot) to a set of destinations (customers).

[0081] In the area of biology and chemistry, protein folding and molecular similarity problems can benefit from the ReduCon methods and systems in accordance with the present embodiments. Finance, particularly financial investing, can benefit from the ReduCon methods and systems in accordance with the present embodiments. Examples include investment portfolio optimization and portfolio allocation.

[0082] Also, there are several software services, products or toolkits where ReduCon could potentially be added or built in to enhance the applicability and performance of the products and provide improved optimization capabilities. For example, Microsoft Azure QIO is a service offering CPU and FPGA implementations of algorithms such as simulated annealing, parallel tempering, population annealing, tabu search, quantum Monte Carlo, and substochastic Monte Carlo. As many of these algorithms are likely to incorporate Houdayer moves, the ReduCon methods and systems in accordance with the present embodiments could improve the algorithms capabilities. A similar Microsoft offering is 1Qbit (1Qloud).

[0083] Matlab Global Optimization Toolbox (not to be confused with the Matlab Optimization Toolbox), Google OR-Tools, Microsoft Excel, Anylogistix supply chain and logistics software, and Gurobi appear to target similar problems and, even if not presently capable of benefiting from ReduCon methods and systems, may evolve into products that will require the optimization problem solving benefits and advantages provide by the ReduCon methods and systems in accordance with the present embodiments.

[0084] To test the advantages of using the ReduCon methods and systems in accordance with the present embodiments to address such problems, a 0-1 Knapsack problem is studied. Given n items numbered 1 to n , each with associated value v_i and weight w_i , the test is to find which subset of the items to pack into a knapsack with weight capacity W such that the value of the knapsack is maximized. The 0-1 Knapsack problem can be formulated as a linear program as seen in Equation (17):

$$\begin{aligned} &\text{minimize} \quad - \sum_{i=1}^n v_i x_i \\ &\text{subject to} \quad \sum_{i=1}^n w_i x_i \leq W \end{aligned} \tag{17}$$

By adding slack variables, one can easily turn the inequality constraint into an equality constraint, thus recovering a problem of the form of Equation (1) or the form of Equations (6) and (7). In particular, the linear constraint creates a full-connectivity in the underlying graph, which is a perfect use-case for the ReduCon methods and systems in accordance with the present embodiments.

[0085] By comparing different variants of the parallel tempering procedure to understand the benefits of the ReduCon methods and systems in accordance with the present embodiments, the Knapsack problem can more thoroughly be investigated and

the methods and systems in accordance with the present embodiments can methodically be tested. The variants are changes in the Monte-Carlo moves used within parallel tempering to explore the state space. The Monte-Carlo moves used are single-flips combined with Houdayer moves, and single-flips combined with ReduCon Houdayer moves, in order to understand how ReduCon is able to enhance the results obtained by only using the Houdayer move. Moreover, the parallel tempering algorithm is run for 2500 iterations using 30 temperatures spaced geometrically between 0.001 and 21, where at each temperature two configurations are simulated. The algorithm works as follows: at every iteration, the two configurations at each temperature are first altered by a single-flip independently. Then, for each temperature, the Houdayer (or ReduCon Houdayer) move is performed on the pair at that temperature. Also, only the energy of the two configurations at the lowest temperature are kept track of.

[0086] Ten instances with $n = 100$ items were chosen with weights w_i chosen uniformly at random in the set $\{1, \dots, 10\}$, values given by $v_i = i$, and weight capacity given by $W = 20$. For each instance, the optimal solution was computed via dynamic programming (since the problem size is small) in order to compare our results. Experiments were performed as described hereafter, where for each instance, the parallel tempering scheme described hereinabove was used.

[0087] Since the optimum value is known in this example, the relative error for each run was computed at each iteration to see its evolution, noting that given some target value v and its approximation $\tilde{v}$, the relative error is defined as $\left| \frac{v - \tilde{v}}{v} \right|$. The mean and standard deviation of the relative errors computed over the 10 instances were then plotted in order to visualize the evolution of the average relative error. Results of the multiple aggregated parallel tempering procedures for different types of Monte-Carlo moves are shown in a graph 1300 in FIG. 13 where the relative error for optimal

solutions computed by Houdayer moves 1310 and the relative error for optimal solutions computed by the ReduCon Houdayer move 1320 are plotted in the graph 1300. For each move, the mean relative error together with its standard deviation was plotted as a function of the iteration number.

[0088] Looking at the graph 1300, some interesting behaviours can be observed. As expected, since the Knapsack has an underlying fully-connected graph, the performance of the single-flips with the Houdayer move is the worst. It is in fact the same as if no Houdayer move was used at all. The most striking result is the outcome of the ReduCon Houdayer move which was able to reduce the relative error by orders of magnitude and even reach the optimal solution, showing the benefit of the ReduCon methods and systems in accordance with the present embodiments.

[0089] While the utility of the ReduCon strategy in the context of Houdayer Monte-Carlo moves has been discussed, at its core the ReduCon methods and systems in accordance with the present embodiments is a scheme which allows one to reduce the density of certain types of interaction in a system by introducing more variables (i.e., “register” variables). Thus, it is clear that the ReduCon methods and systems in accordance with the present embodiments can be applied in other situations where the ultimate objective is not to improve the success of Houdayer moves.

[0090] Specifically, the ReduCon methods and systems in accordance with the present embodiments may be applied when a high graph density is potentially induced by an optimization constraint, such as that of Equation (6). Optimization problems of this nature are very common (e.g., the Knapsack problem discussed above), and in recent years there has been much interest in solving them in the Ising/QUBO formulation. At the same time, there has emerged a generation of specialized CMOS classical hardware designed specifically for running highly parallel implementations of physics-based

algorithms, such as simulated annealing and parallel tempering. Optics-based platforms may also be used to solve complex combinatorial optimization problems, leveraging the natural physical behaviour of the system to find its lowest energy state.

[0091] In constructing such hardware, it may be challenging to fully interconnect many thousands of nodes or qubits, resulting in limitations to the scalability of such devices to larger sizes. For the classes of problem where ReduCon may be applied, the ReduCon methods and systems in accordance with the present embodiments could be used to design specialized hardware with a lower density of connectivity nodes, potentially overcoming scalability issues. Since constrained optimization problems similar to the Knapsack problem are of significant economic value, there may be a compelling case for constructing hardware designed specifically to solve them. Thus, ReduCon systems in accordance with the present embodiments implemented in hardware products designed specifically for optimization resolution, and even being designed for specific optimization problem(s), can provide advantages and benefits as discussed herein.

[0092] Thus, it can be seen that the present embodiments provide systems and methods for a novel procedure designed to reduce the connectivity of graphs underlying Ising/QUBO problems. In doing so, new avenues are opened to use the Houdayer cluster Monte-Carlo algorithm, enabling one to reach lower energy levels in settings where it would previously fail. In accordance with the present embodiments, systems and methods are provided which enable improved reduction in the density of certain types of interaction in a system by introducing more variables (i.e., “register” variables), those variables, in some instances, forming barriers (i.e., “register barriers”) to allow connected components to not join together and thereby reducing connectivity. Since the ReduCon methods and systems in accordance with the present embodiments

describe a relatively general procedure, there are multiple choices to be made when employing them. In particular, schemes in accordance with the present embodiments yield groupings of variables which empirically increase the chance of success of the Houdayer move.

[0093] Overall, the methods and systems in accordance with the present embodiments go hand in hand with the Houdayer move and, thus, can be used in any application where the Houdayer move is used, such as the behaviour of parallel tempering on a specific Knapsack instance, an optimization problem where the usual Houdayer move is inefficient. In addition, the ReduCon methods and systems in accordance with the present embodiments allows a variety of complex problems to enjoy the benefits of the Houdayer move, resulting in a better optimization.

[0094] While exemplary embodiments have been presented in the foregoing detailed description of the present embodiments, it should be appreciated that a vast number of variations exist. It should further be appreciated that the exemplary embodiments are only examples, and are not intended to limit the scope, applicability, operation, or configuration of the invention in any way. Rather, the foregoing detailed description will provide those skilled in the art with a convenient road map for implementing exemplary embodiments of the invention, it being understood that various changes may be made in the function and arrangement of steps and method of operation described in the exemplary embodiments without departing from the scope of the invention as set forth in the appended claims.

CLAIMS

What is claimed is:

1. A computer-implemented method for transformation of an optimization problem to facilitate its resolution, the method comprising:

casting the optimization problem into a quadratic unconstrained binary model;

and

transforming the optimization problem into an optimization problem having the quadratic unconstrained binary model with reduced connectivity, wherein transforming the optimization problem comprises:

partitioning decision variables in the quadratic unconstrained binary model into two or more groups, each of the two or more groups comprising at least one decision variable node; and

introducing a register variable node between adjacent pairs of the at least one decision variable node in the two or more groups to hold partial values of a sum in a linear constraint to form the optimization problem with reduced connectivity.

2. The method in accordance with Claim 1, wherein transforming the optimization problem comprises forming a graphical visualization transformation of the quadratic unconstrained binary model before the partitioning step.

3. The method in accordance with Claim 2, wherein the graphical visualization transformation comprises a graph network having nodes and edges.

4. The method in accordance with Claim 3, wherein the nodes are qubits.

5. The method in accordance with any of the preceding claims, wherein casting the optimization problem comprises casting the optimization problem in a form of a quadratic unconstrained binary model by turning linear equality constraints into quadratic penalties of the quadratic unconstrained binary model.
6. The method in accordance with any of the preceding claims, wherein introducing a register node comprises introducing two or more register variable nodes to form the optimization problem with reduced connectivity, each of the two or more register variable nodes introduced between adjacent pairs of the at least one decision variable node in the two or more groups to hold the partial values of a sum in a linear constraint, and wherein transforming the optimization problem comprises forming a register variable edge between adjacent pairs of the introduced two or more register variable nodes.
7. The method in accordance with any of the preceding claims, wherein each of the one or more groups are different from one another.
8. The method in accordance with any of the preceding claims, wherein each register variable node acts as a buffer during an overlap process.
9. The method in accordance with any of the preceding claims, wherein the quadratic unconstrained binary model comprises a quadratic unconstrained binary

optimization (QUBO) model or an Ising model based on a range of variables in the quadratic unconstrained binary model.

10. The method in accordance with any of the preceding claims, wherein at least one register variable node comprises a plurality of register variables.

11. The method in accordance with any of the preceding claims, wherein the partitioned decision variables are interconnected with one another.

12. The method in accordance with any of the preceding claims, further comprising removing a last decision variable node from the configuration of more than one decision variable nodes in the reduced connectivity optimization problem.

13. The method in accordance with any of the preceding claims, further comprising performing a Houdayer move on the reduced connectivity optimization problem thus resulting in a new optimization problem, the method further comprising casting the new optimization problem into a new quadratic unconstrained binary model and transforming the new quadratic unconstrained binary model by the steps of partitioning decision variables and introducing a register variable.

14. The method in accordance with Claim 13, wherein performing the Houdayer move on the reduced connectivity optimization problem comprises:

converting the two or more groups comprising the at least one decision variable node in an underlying graph of the reduced connectivity optimization problem to a

corresponding two or more groups comprising at least one decision variable node in an underlying graph of the new optimization problem;

performing the Houdayer move on the underlying graph of the new optimization problem to generate a new set of two or more groups comprising at least one decision variable node; and

converting the new set of two or more groups comprising at least one decision variable nodes to a corresponding one or more groups comprising at least one decision variable node in the underlying graph of the reduced connectivity optimization problem.

15. The method in accordance with any of the preceding claims, wherein the step of partitioning decision variables comprises partitioning the decision variables in the quadratic unconstrained binary model into one or more groups in a quantum system.

16. A system for transformation of an optimization problem to facilitate resolution of the optimization problem, the system comprising:

one or more processors; and

storage means connected to the one or more processors, the storage means comprising instructions for controlling the one or more processors to:

cast the optimization problem into a quadratic unconstrained binary model; and

transform the optimization problem into an optimization problem having the quadratic unconstrained binary model with reduced connectivity, wherein transforming the optimization problem comprises instructions to control the processor to:

partition decision variables in the quadratic unconstrained binary model into two or more groups, each of the two or more groups comprising at least one decision variable node; and

introduce a register variable node between adjacent pairs of the at least one decision variable node in the two or more groups to hold partial values of a sum in a linear constraint to form the optimization problem with reduced connectivity.

17. The system in accordance with Claim 16, wherein the instructions control the one or more processors to transform the optimization problem by forming a graphical visualization transformation of the quadratic unconstrained binary model before partitioning the decision variables.

18. The system in accordance with Claim 17, wherein the graphical visualization transformation comprises a graph network having nodes and edges.

19. The system in accordance with Claim 18, wherein the nodes are qubits.

20. The system in accordance with any of Claims 16 to 19, wherein the instructions to control the one or more processors to cast the optimization problem comprise instructions to control the one or more processors to cast the optimization problem in a form of a quadratic unconstrained binary model by turning linear equality constraints into quadratic penalties of the quadratic unconstrained binary model.

21. The system in accordance with any of Claims 16 to 20, wherein the instructions to control the one or more processors to introduce the register variable node comprise instructions to control the one or more processors to introduce more than one register variable node, each of the more than one register variable node introduced between different adjacent pairs of the at least one decision variable node in the two or more partitioned groups, and wherein transforming the optimization problem comprises forming a register variable edge between adjacent pairs of the more than one register variable nodes introduced.

22. The system in accordance with any of Claims 16 to 21, wherein each of the one or more configurations are different from one another.

23. The system in accordance with any of Claims 16 to 22, wherein the register variable node acts as a buffer during an overlap process.

24. The system in accordance with any of Claims 16 to 23, wherein the quadratic unconstrained binary model comprises a quadratic unconstrained binary optimization (QUBO) model or an Ising model based on a range of variables in the quadratic unconstrained binary model.

25. The system in accordance with any of Claims 16 to 24, wherein each of the at least one register variable node comprises a plurality of register variables.

26. The system in accordance with any of Claims 16 to 25, wherein the partitioned decision variables are interconnected with one another.

27. The system in accordance with any of Claims 16 to 26, wherein the instructions to control the one or more processors further comprise instructions to remove a last decision variable node from the configuration of more than one decision variable nodes in the reduced connectivity optimization problem.

28. The system in accordance with any of Claims 16 to 27, wherein the instructions to control the one or more processors further comprises instructions to perform a Houdayer move on the reduced connectivity optimization problem thus resulting in a new optimization problem, the system further comprising instructions to control the one or more processors to cast the new optimization problem into a new quadratic unconstrained binary model and to transform the new quadratic unconstrained binary model by the steps of partitioning decision variables and introducing a register variable.

29. The system in accordance with Claim 28, wherein the instructions to control the one or more processors to perform the Houdayer move on the reduced connectivity optimization problem comprise instructions to control the one or more processors to:

convert the two or more groups comprising the at least one decision variable node in an underlying graph of the reduced connectivity optimization problem to a corresponding two or more groups each comprising at least one decision variable node in an underlying graph of the new optimization problem;

perform the Houdayer move on the underlying graph of the new optimization problem to generate a new set of two or more groups each comprising at least one decision variable node; and

convert the new set of two or more groups comprising at least one decision variable node to a corresponding one or more groups each comprising at least one decision variable node in the underlying graph of the reduced connectivity optimization problem.

30. The system in accordance with any of the Claims 16 to 29, wherein the instructions to control the one or more processors to partition decision variables comprises instructions to control the one or more processors to partition the decision variables in the quadratic unconstrained binary model into one or more groups in a quantum system.