



Quantum Bridge Analytics II: QUBO-Plus, network optimization and combinatorial chaining for asset exchange

Fred Glover¹ · Gary Kochenberger² · Moses Ma³ · Yu Du²

Received: 2 November 2019 / Revised: 7 November 2020
© Springer-Verlag GmbH Germany, part of Springer Nature 2021

Abstract

Quantum Bridge Analytics relates to methods and systems for hybrid classical-quantum computing, and is devoted to developing tools for bridging classical and quantum computing to gain the benefits of their alliance in the present and enable enhanced practical application of quantum computing in the future. This is the second of a two-part tutorial that surveys key elements of Quantum Bridge Analytics and its applications. Part I focused on the Quadratic Unconstrained Binary Optimization (QUBO) model which is presently the most widely applied optimization model in the quantum computing area, and which unifies a rich variety of combinatorial optimization problems. Part II (the present paper) introduces the domain of QUBO-Plus models that enables a larger range of problems to be handled effectively. After illustrating the scope of these QUBO-Plus models with examples, we give special attention to an important instance of these models called the Asset Exchange Problem (AEP). Solutions to the AEP enable market players to identify exchanges of assets that benefit all participants. Such exchanges are generated by a combination of two optimization technologies for this class of QUBO-Plus models, one grounded in network optimization and one based on a new metaheuristic optimization approach called combinatorial chaining. This combination opens the door to expanding the links to quantum computing applications established by QUBO models through the Quantum Bridge Analytics perspective. We show how the modeling and solution capability for the AEP instance of QUBO-Plus models provides a framework for solving a broad range of problems arising in financial, industrial, scientific, and social settings.

Keywords Quantum Bridge Analytics · Combinatorial chaining · Network optimization · Asset exchange technology · Quantum computing · Blockchain

Extended author information available on the last page of the article

1 Introduction

Quantum Bridge Analytics is devoted to developing tools for bridging classical and quantum computing to gain the benefits of their alliance in the near term and enable enhanced practical application of quantum computing in the future.

As observed in Part I of this tutorial (Glover et al. 2019), the Quadratic Unconstrained Binary Optimization (QUBO) model has an important role in Quantum Bridge Analytics by unifying a rich variety of combinatorial optimization problems and becoming at present the most widely applied optimization model in the quantum computing area.

In Part II (the present paper) we consider applications called QUBO-Plus problems motivated by the classical QUBO formulation that embrace larger classes of problems, and that also make it possible to solve certain subclasses of QUBO problems more effectively. Details of the QUBO-Plus model, including its formulation and illustrative applications, are discussed in Sect. 2.

To underscore the importance of identifying problems that can be treated as QUBO-Plus applications, we give special attention in this paper to a problem class called the Asset Exchange Problem (AEP) which is motivated by developments in blockchains that relate to finding exchanges among investors that are mutually beneficial to all participants. We first describe a QUBO model for a special instance of the AEP and then adopt the QUBO-Plus perspective to consider the relevance of characterizing the AEP domain in a more general form. To further motivate our focus on the AEP we discuss a range of applications both within and beyond blockchains where this class of problems is important. As we demonstrate in the sections that follow, solutions to the AEP enable market players to identify and profit from exchanges of assets that benefit all participants—exchanges that, in game theory terminology, constitute a positive sum game. This provides a mechanism for facilitating exchanges customarily carried out through mechanisms of money, interest, and middlemen by serving instead the blockchain goal of disintermediation to remove or reduce reliance on intermediaries. The resulting modeling and solution process simultaneously afford a link between the applications of classical and quantum computing that are envisioned to be increasingly relevant as the quantum computing area becomes more mature.

We introduce two main hubs for solving AEP models, the first consisting of a mathematical formulation yielding a network optimization model for a basic version of the AEP and the second consisting of a metaheuristic optimization framework called combinatorial chaining that augments the network model to make it possible to derive high quality solutions to more complex instances of the AEP, notably including instances encountered in a wide variety of real world applications.

These developments derive special relevance within the context of Quantum Bridge Analytics, which offers gains by bridging the gap between classical and quantum computational methods and technologies. As observed in the 2019 Consensus Study Report titled *Quantum Computing: Progress and Prospects* (National Academies 2019), quantum computing will remain in its infancy for perhaps another decade, and in the interim “formulating an R&D program with the aim of developing commercial applications for near-term quantum computing is critical to the health of

the field.” The report further notes that such a program will rely on developing “hybrid classical-quantum techniques.” Innovations that underlie and enable these hybrid classical-quantum techniques, which are the focus of Quantum Bridge Analytics, provide a fertile catalyst for introducing the QUBO-Plus applications in the AEP domain.

Additional links to the QBA theme are provided in Kochenberger and Ma (2019) who observe that QUBO and QUBO-Plus models give rise to a variety of formulations for portfolio optimization, and these in turn yield a natural basis for integrating classical and quantum computing via the Asset Exchange Problem. Portfolio optimization has a prominent role in the AEP when the assets under consideration involve those customarily incorporated into the portfolio domain. The AEP goes further, both in the portfolio domain and others, by linking the holders of multiple portfolios in a network of cooperative optimization. This establishes a natural alliance with QUBO-Plus models whose solutions identify desirable assets for different participants.

After introducing the general representation of QUBO-Plus models in the next section and providing examples of applications related to the theme of the present paper, in succeeding sections we provide a mathematical optimization model for a basic instance of the AEP and then show how the model can be transformed into a network optimization model, thus laying the foundation for exploiting more complex variants of the AEP.

A note on terminology: we use the term “exchange” rather than “swap” because a swap typically refers to an exchange involving only two items or two participants, and “multiple swaps” refer to a collection of pairwise exchanges, in contrast to an integrated process that requires the coordination among all participants for its execution.

The most developed literature on exchanges occurs for the traveling salesman problem, where the term *k-opt* refers to an exchange that removes k edges from a tour and replaces them by k other edges so that the resulting configuration continues to be a tour [Hamiltonian cycle; see, e.g., Helsgaun (2000) and Helsgaun (2009)]. The traveling salesman procedures that come closest to the process of combinatorial chaining are the ejection chain approaches that have been applied to TSPs and other combinatorial optimization problems (Glover 1996; Rego and Glover 2006; Yagiura et al. 2006, 2007; Rego et al. 2016).

The blockchain literature refers to exchanges called *atomic swaps* (also known as cross-chain trading). As elaborated subsequently, these exchanges arise when two parties who want to share their cryptocurrencies execute an exchange by means of Hashed Timelock Contracts (or HTLCs) as a mechanism to make the transaction secure (Nolan 2013; Fitzpatrick 2019). Combinatorial chaining makes it possible to generalize these swaps to exchanges involving multiple actors.

Combinatorial chaining and the Asset Exchange Problem are to be differentiated from the problem and methods arising in combinatorial auctions where swaps are sought to exchange pairs of buy/sell-orders in futures markets (Winter et al. 2011; Müller et al. 2017). An interesting area for future investigation would be to determine if the combinatorial chaining approach could likewise be applied in the setting of combinatorial auctions to enable auctions involving greater numbers of participants.

The remainder of this paper is organized as follows. Section 2 introduces QUBO-Plus models that provide computationally important alternatives to standard QUBO models. Examples of asset exchange applications are given in Sect. 3 to set the stage for later more extensive and technical discussions. Section 4 provides the fundamental mathematical formulation of the basic AEP problem, and shows how to transform this formulation into a network optimization model. Section 5 characterizes the structure of combinatorial chaining in reference to this basic network model, followed by introducing more advanced processes in Sect. 6 for joining network optimization and combinatorial chaining with metaheuristic analysis to address more complex instances of asset exchanges. The paper concludes with a summary of the key notions and their implications in Sect. 7.

2 QUBO-Plus models and the asset exchange problem

The classical QUBO model is expressed as follows.

$$\text{QUBO: Minimize/Maximize } y = x^t Qx$$

where x is a vector of binary decision variables and Q is a square matrix of constants.

The term “QUBO-Plus” refers to a class of models that augment the preceding standard QUBO representation by introducing important constraints separately from the QUBO model, enabling them to be treated by algorithms specially designed to handle these special constraints. This contrasts with the standard approach that seeks to merge such constraints with the QUBO model by attaching weights to them to create a modified form of the Q matrix as described in the Part I tutorial. Many problems have special constraints that could be modeled in pure QUBO form but may afford advantages from both a computational and “model transparency” point of view by embodying them in a QUBO-Plus model. By keeping these constraints separate from the QUBO formulation, and developing a special algorithm that handles the resulting QUBO-Plus problem, it is possible to solve these problems more efficiently and effectively than by attempting to create a “transformed” QUBO model that folds the constraints into the Q matrix.

Computational studies (Du et al. 2020) document that QUBO-Plus models often deliver superior performance, relative to transformed QUBO alternatives, in terms of solution quality and solution time, and permit larger problems to be solved. Such QUBO-Plus models also provide a transparent reminder of the special constraint(s) that are otherwise lost in a transformed QUBO representation.

While the variety of QUBO-Plus models is substantial and application dependent, we catalog some commonly encountered special cases in the following list. In the cases highlighted below, the reference to setting a variable to 1 can encompass a variety of applications by a correspondence with selecting a particular item from a collection of projects, investments, assets, facilities, locations, buildings, plans, medical treatments, architectural designs, itineraries, etc. In such settings, the special constraint(s) defining the “Plus” component of the QUBO-Plus model embody a key problem feature earmarked for special treatment.

2.1 Common QUBO-Plus model types

- (1) Exact cardinality constrained QUBO problems: requiring an exact number of variables to be set to 1.
- (2) Bounded cardinality constrained QUBO problems: requiring a lower bound and an upper bound on the number of variables set to 1.
- (3) Multi-assignment QUBO problems: requiring disjoint sets of variables to sum to 1.
- (4) Multi-allocation QUBO problems: requiring disjoint sets of variables to sum to specified constants (that may differ from 1).
- (5) QUBO packing problems: requiring sums of variables to be less than or equal to specified constants.
- (6) QUBO covering problems: requiring sums of variables to be greater than or equal to specified constants.
- (7) QUBO problems combining (5) and (6) (also called bounded multi-allocation QUBO problems).
- (8) QUBO knapsack problems: requiring a weighted sum of variables to be less than or equal to a specified constant.
- (9) Multi-knapsack QUBO problems: requiring weighted sums of variables to be less than or equal to specified constants.
- (10) Generalized covering QUBO problems: requiring weighted sums of variables to be greater than or equal to specified constants.

With appropriate modifications, modern QUBO solvers can be customized to produce solutions that satisfy the explicit “Plus” constraints while optimizing the quadratic objective function.

We remark that QUBO-Plus models of type (1) arise naturally in the context of the well-known Maximum Diversity problem and also in portfolio optimization problems where a pre-specified number of assets must be chosen. These models are generalized by those of type (2) which apply to broader settings. QUBO-Plus type (3) models, with their assignment constraints, have many important applications and are further noteworthy for having natural connections to certain types of network problems. In these network-related problems, members of certain groups or sub-groups are assigned to members of other groups with the goal of optimizing some measure that describes the overall effectiveness of the assignments. Problems of this type have a link to the AEP problem via their network-related component.

We describe specific models below that illustrate these connections.

Model 1. While applications of all ten QUBO-Plus type models are found in practice, Type (3) models with their assignment constraints are particularly noteworthy due to their connection to various forms of clustering and applications such models accommodate. Consider for example, an investment setting where $x_{ij} = 1$ if investor i is assigned to cluster (class of investments) j , and the constraints are $\sum(x_{ij}: \text{over all } j) = 1$ for each investor i ensure that each investor is assigned. A variation of this type of application is where particular investments are assigned to specific investment classes. Each of these applications involves considerations that are relevant to the AEP problem,

although they fall short of capturing a variety of additional elements of the AEP as we subsequently indicate.

Model 2. Graph coloring problems present additional applications for QUBO-Plus type (3) models where a color must be assigned to each node in the graph (the assignment constraints) while adjacent nodes are required to receive different colors. The adjacency constraints can be folded into the Q matrix and the node assignment constraints can be imposed traditionally rather than by penalties in the objective function.

The coloring terminology takes on a practical meaning by equating colors with labels used to categorize objects (people, institutions, groups, products, processes, etc.). Nodes that are adjacent (joined by an edge), can be viewed in a context where edges between nodes may be interpreted, for example, to mean that the associated objects are dissimilar, hence a coloring will categorize objects so that dissimilar objects fall in different categories. Minimizing the number of colors results in minimizing the number of categories needed to differentiate the objects. Other interpretations of the adjacency relationship lead to additional applications.

More formally, for a graph $G = (V, E)$ with n vertices, the Minimum Sum Coloring Problem (MSCP) (Kubicka and Schwenk 1989) seeks to find a coloring such that the sum of all the colors over all vertices is minimized. If we define $x_{ik} = 1$ if color k is assigned to vertex i , and we seek a coloring using at most K colors, then the adjacency conditions are satisfied by $x_{ik} + x_{jk} \leq 1$ for all $(i, j) \in E$ and all $k \in \{1, \dots, K\}$. For a positive scalar penalty, P , these constraints can be imposed via penalties of the form $Px_{ik}x_{jk}$ to be added to objective function. Proceeding in this fashion yields the penalized objective function:

$$\text{Minimize } \sum_{i=1}^n \sum_{k=1}^K k * x_{ik} + P \left(\sum_{(i,j) \in E} \sum_{k=1}^K x_{ik}x_{jk} \right)$$

which can be re-written as in the form of $x^t Qx$.

Including the node assignment constraints without taking them into the objective function, we have the Minimum Sum Coloring Problem in the following form:

$$\begin{aligned} \text{Minimize } & \sum_{i=1}^n \sum_{k=1}^K k * x_{ik} + P \left(\sum_{(i,j) \in E} \sum_{k=1}^K x_{ik}x_{jk} \right) \\ \text{s.t. } & \sum_{k=1}^K x_{ik} = 1 \quad \forall i \in \{1, \dots, n\} \end{aligned}$$

which is a QUBO-Plus model:

$$\begin{aligned} \text{Minimize } & x^t Qx \\ \text{s.t. } & \sum_{k=1}^K x_{ik} = 1 \quad \forall i \in \{1, \dots, n\} \end{aligned}$$

Model 3. A practical application involving assignment constraints more closely related to the problems we treat in this paper involves exchanges to re-balance a set of account assignments for account executives in a large organization. For this example, assume that the company in question has K account executives, each responsible for managing a set of accounts. Currently the firm has $P > K$ accounts of varying size in terms of annual revenues. Denote the annual dollar amount of account p by d_p dollars. As business has grown over the past few years, the total dollar amount of business managed by a given account executive has grown in an uneven fashion to produce considerable disparity in the total volume of business managed by a given executive.

To address this issue, top management wants to re-assign accounts to the various account executives to re-balance the system and make the differences in total annual book of business between account executives as small as possible.

This problem can be modeled and solved by a QUBO-Plus model with assignment constraints as follows:

Let $x_{pk} = 1$ if account p is assigned to executive k ; zero otherwise. Then, the total annual volume of business managed by executive k is

$$\text{total}_k = \sum_{p=1}^P d_p x_{pk} \quad k = 1, \dots, K$$

Our objective is to make assignments of accounts to executives so that each account gets assigned to an executive while minimizing the squared deviations from one executive to another. That is, our objective function is:

$$\text{Minimize deviation} = (\text{total}_1 - \text{total}_2)^2 + (\text{total}_1 - \text{total}_3)^2 + \dots + (\text{total}_{K-1} - \text{total}_K)^2$$

Substituting the definition of total_p into the above and writing the objective function in matrix form, we get the QUBO-Plus model with assignment constraints:

QUBO-Plus model:

$$\begin{aligned} &\text{Minimize } x^t Q x \\ &\text{s.t. } \sum_{k=1}^K x_{pk} = 1 \quad \forall p \in \{1, \dots, P\} \end{aligned}$$

where Q is the square, symmetric matrix that results from collecting terms and x is the vector of x_{pk} variables relabeled with a single subscript. The constraints ensure that each account gets assigned to one of the executives and the objective function ensures that the differences in total account values from one executive to another are as small as possible. Solving this model will result in a new set of account assignments that re-balance the system.

Our next example illustrates a more general type of application that corresponds to a QUBO-Plus model of type (4), and likewise has features in common with those we address in the AEP problem. In this case, we have selected an application relevant to responding to an outbreak of an epidemic.

Model 4. The goal of this model is to determine an optimal allocation of testing kits, as in the context of virus detection, to a population of n people whose members may or may not be infected. Suppose q_i = the estimated value of having person i tested, and q_{ij} = the additional value of having both i and j tested (beyond the value of $q_i + q_j$)—as where these individuals interact with different groups of people and it is desirable not to limit testing to those who interact within the same group. The q_{ij} coefficients can be made larger if the groups that person i and person j interact with are high risk groups.

The objective is to maximize the total value of the people tested. This can be modeled as a QUBO-Plus problem where the number of people tested equals KitsAvailable, the number of test kits available on a given day.

Define $x_i = 1$ if person i is given the test; 0 otherwise. Then the cardinality constrained QUBO-Plus model is obtained by:

$$\begin{aligned} \text{Maximize Total Testing Value} &= \sum_{i=1}^n q_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n q_{ij} x_i x_j & (C1) \\ \text{subject to} & \sum_{i=1}^n x_i = \text{KitsAvailable} \end{aligned}$$

This basic model incorporating constraint (C1) can readily be enriched in a variety of ways. For instance, suppose each person i belongs to a Group k indexed by $k = 1, \dots, K$, where members of Group k are identified because they interact or because they are a geographical community or have other features in common considered likely to influence their risk. (These groups may also be identified by a QUBO clustering algorithm.) This results in constraints of the form

$$\sum_{i \in I_k} x_i = V_k \text{ for } k = 1, \dots, K \quad (C2)$$

where I_k denotes the individuals in group k and V_k denotes the number of kits to be allocated to group k .

Here the sum of the V_k values equals KitsAvailable. The V_k values may be proportional to the sizes of the Groups k or may be skewed by the estimated riskiness of each group as a whole.

The foregoing problem with constraints (C1) and (C2) is easily formulated as a QUBO problem by taking the constraints into the objective function as illustrated in Part 1 of this tutorial (Glover et al. 2019), but can be solved more effectively by a special QUBO-Plus algorithm specifically designed for this class of QUBO applications. Constraints (C1) and (C2) are instances of more general constraints that arise in the AEP problem and are common in many network formulations.

2.2 Differentiation among QUBO-Plus models

In essence, we have three types of QUBO-Plus models. The first type consists of those for which a QUBO formulation can be readily constructed by incorporating certain constraints into the objective function, but the problem can be solved more effectively by a special approach that keeps these constraints separate. The second type consists of a binary choice “logical” problem where a QUBO formulation is exceedingly difficult to construct, yet where again we can develop an effective framework for solving it motivated by the concepts developed for representing and solving QUBO problems. The third type is an extension of the first two, arising in response to practical applications that embody highly exploitable problem structures, such as those in the domain of network-related models which are accompanied by additional combinatorial conditions that take them beyond classical analysis, but that can nevertheless be susceptible to tailored algorithms based on the principles that have produced the most effective QUBO methods.

The first type of QUBO-Plus model is illustrated by the QUBO-Plus formulations above. The second type of model includes problems that have binary network-related formulations, where we can exploit the fact that we can represent basic instances of these problems as QUBO problems and QUBO-Plus problems of the first type. The Asset Exchange Problem at the focus of this paper belongs to the third category. We will show that we can capture several essential features of this application within a network optimization model, although one that is exceedingly large and computationally demanding. We will disclose how this structure can alternatively be made susceptible to an approach called combinatorial chaining, where we employ a strategy shared with the most effective QUBO algorithms, which iteratively identifies sub-structures that can be successfully to exploited to yield progressive improvement. A basic (rudimentary) form of combinatorial chaining is presented in Sect. 5 for simpler AEP problem instances. Then in Sect. 6 we describe modifications that give rise to advanced forms of combinatorial chaining that handle more complex AEP formulations.

3 Preliminary examples of the asset exchange problem

As previously intimated, the Asset Exchange Problem (AEP) arises in a variety of contexts, spanning applications in financial investment, resource allocation, economic distribution and collaborative decision making. Our approach to solving this problem is based on a form of cooperative optimization, where multiple parties with complex criteria collaborate as well as compete for resources. This could apply to algorithms for distributing packages between trucks in a delivery network, or dynamic switching to alternative sorting facilities. Or it could apply to collaborative bidding processes for complex multi-criteria contracts or decentralized cooperative group optimization for multi-criteria investment cryptocurrency portfolios. This is quite distinct from traditional portfolio optimization, as with a hedge fund that typically seeks to mitigate risk by diversification with some investments that are negatively correlated.

In the cooperative group optimization setting, our approach generalizes processes that seek atomic exchanges of baskets of fungible tokens or securities by yielding

exchanges at a higher combinatorial level. Normally, a financial institution that wishes to execute a large basket of trades, in a way that mitigates execution risk by having an intermediary, can take the basket into its inventory and unwind the trades on its own. Thus, instead of revealing specific information about the assets in the basket, knowledge which could be exploited, the institution and banks can conduct a “zero-knowledge” protocol to effect basket trades. However, this protocol still requires trust in those institutions providing the service. The proposed new approach uses both simple and complex combinatorial exchanges to optimize all parties engaged in the multi-party optimization effort.

The progenitor of such an approach has emerged and is being tested in the cryptocurrency world—this is known as a *cross-chain atomic swap*. This is where two parties own tokens in separate cryptocurrencies, and want to exchange them without having to trust a third party or a centralized exchange. However, by extending this model and enabling complex multi-party swaps, splits and aggregations, we can effect full spectrum combinatorial trading to provide *trustless algorithmic liquidity* without requiring even the normal underlying reserve trading currency.

The simplest instance of such a system is a marketplace of three portfolios. In this market, Portfolio A has asset X and wishes to own some asset Z, Portfolio B has asset Z but wishes to acquire only asset Y, and Portfolio C has asset Y and wishes to own some asset X. In a traditional exchange, participants would exchange what they have for the underlying reserve settlement currency, and then purchase what they want. This would entail two transaction fees per portfolio. Alternatively, using cross chain atomic swaps, the parties would never make any transactions whatsoever, as the global optimal cannot be reached via pairwise swap transactions.

By enabling all potential complex exchanges, splits and aggregations, for N portfolios, any market could increase its global utility. However, the computational complexity of this type of complex combinatorial exchange trading is NP-complete. By using a multi-attribute trade matching system that includes the unspoken goals of the parties, which are the “utility functions” of the parties, it is possible to find Pareto-efficient exchange solutions—referring to the game theory concept of a strategy that cannot be made to perform better against one opposing strategy without performing less well against another.

Additionally, the inclusion of constraints increases the complexity of the problem. For example, if the system determines that diversification is required, then a constraint could be added that limits which types of assets could be included in the diversification target. Only assets that have been rated by a rating agency or analyst, for example, as better than a “B” rating, could be included to modify the optimization. A continuous approach would assign each rating a numerical value, and blend that with volatility metrics, volume data, social impact scores, and even the user’s personal pet peeves—to enable a multi-objective approach to optimize both individual and multiple portfolios.

In the future, the user will require the ability to enter or modify both market orders (with fixed prices) and limit orders (with variable prices). As we transition from market orders to limit orders, this will help to expand utility expression, and it can become appropriate to add constraints to help identify price improvement opportunities - allowing a combinatorial exchange to operate for a share of price improvement, rather than charging transaction fees. Just as Bitcoin promises “zero cost transactions”, this could

provide a model for “zero cost exchanges” that provide the appearance of negative transaction cost given a disparity of utility functions. In Sect. 6 we discuss the use of priorities to address such considerations.

The current model for the most effective form of exchange is the double-sided exchange, a system in which both buyers and sellers provide bids for matching via the exchange. A central controlling system matches the sell bids with the buy bids, yielding matched buy bids and matched sell bids in response thereto so that allocations of the matched buy bids and the matched sell bids maximize the throughput of the exchange. Combinatorial exchanges using cooperative optimization could potentially lay a foundation for the next evolutionary step in market exchange protocols, moving from double sided trading using a reserve currency to something more general that encompasses new forms of economic transactions.

Double sided exchanges are used to trade goods, services, or other things of value, including network bandwidth trading, financial-instruments trading, transportation logistics, pollution-credit trading, electric power allocation, and so on. However, to make double sided exchanges work, they require fungibility. And so, varying levels of quality, that describe for example the quality of crude oil, are lumped into fungible categories of sulfur content, gravity, etc. This further suggests the possibility that combinatorial exchanges could reflect multi-attribute trading more effectively, allowing traders to work with greater accuracy in pricing.

Combinatorial exchanges can likewise be used for handling non-fungible assets. As long as people are willing to assign value to objects to be traded, combinatorial exchanges can provide a basis to get people what they want. Suppose User A wants to sell a vacation timeshare he or she is tired of, for a certain collectible car with roughly the same value. There are no matches as both are relatively illiquid markets and it could take several months or require a significant discount to find buyers. However, there could be a User B who has exactly the car A wants, but doesn't want a timeshare, and instead wants a diamond necklace. Now if there is a jeweler C who would find that timeshare exciting, and willing to create a custom necklace to B's liking, the system could enable algorithmic liquidity by joining all three into a complex transaction.

Moving toward a more general example, A and B's assets most likely have different values. If there is no jeweler willing to make just the right necklace, the value exchange would not add up. Two parties would likely need to add or accept part of the value in cash. However, with the inclusion of a special user D, who is willing to inject cash and accept a partial tokenized share of that collectible car or real estate, the complex transaction becomes possible. We call this special user a “decentralized market maker” who would require a modest premium to compensate for enabling greater liquidity. That token share could be sold at a later time, hence it is an offer to sell cash for time.

Additional connections to blockchains via decentralized market making are described in “Appendix 1”.

One last note concerns the potential for quantum computing in this arena. In general, present day quantum computers can handle only a very limited number of qubits, representing a small number of asset types, or cooperating portfolios. When quantum computers can offer hundreds of thousands of qubits, with effective partitioning algorithms, combinatorial exchanges will be able to scale to manage real world liquidity needs for applications involving massive numbers of participants and classes of items

to exchange. Until then, quantum computing will enable exchange functionality for only limited and constrained markets, such as for cryptocurrencies. For example, a crypto wallet that holds only a dozen types of crypto would represent a relatively small variable space and could potentially be optimized using a quantum computer. Money was invented to simplify barter, and a quantum exchange based on Pareto-efficient combinatorial exchanges could simplify money.

As noted in Part I, the QUBO model has been adopted as a central focus by the quantum computing community, and by groups that aspire to solve problems by emulating quantum computation with classical hardware. Motivated by the Quantum Bridge Analytics perspective we can enlarge this focus to embrace QUBO-Plus models, and in particular those of the third type discussed in Sect. 2.2. As we will show, this enables us to provide the desired exchanges by identifying combinatorial chaining algorithms that are capable of accommodating variable spaces for AEP models of significantly greater dimension, providing advances in the near term that can be translated into progressively greater advances in the future as quantum computing technology becomes more mature.

4 Mathematical formulations of the AEP

The Asset Exchange Problem has several levels. We start from the most basic level of the AEP, which we call AEP0, defined in reference to a graph $G = (N, E)$, with node set $N = \{1, \dots, n\}$ and edge set $E = \{\{i, j\}, i, j \in N\} \subset N \times N$. Each node $i \in N$ identifies an entity such as an individual or business or institution, and each edge $\{i, j\}$ identifies an *exchange link* between i and j . Let A denote the set of asset types (classes), where elements $\alpha \in A$ can represent classes of tokens in a cryptocurrency application or types of securities in a securities market or categories of commodities in a commodity market, and so forth. In the following we use the term *assets* interchangeably with the term *asset types*.

In AEP0, each node $i \in N$ has a set S_i of assets it can send (i.e., can agree to send) to other nodes and a set R_i of assets it can receive (i.e., can agree to receive) from other nodes. Thus, for example, if $\alpha' \in S_i$ and $\alpha'' \in R_i$, then node i can agree to send asset α' and agree to receive asset α'' from other nodes. More precisely, R_i denotes assets that i desires (considers beneficial) to receive and S_i denotes assets that i is willing to send (in return for obtaining an asset in the set R_i). We say a transfer of asset α from node i to node j is *admissible* if $\alpha \in S_i$ and $\alpha \in R_j$ ($\alpha \in S_i \cap R_j$). We allow only admissible transfers in seeking asset exchanges that benefit all participants.

Define $N_i = \{j \in N : \{i, j\} \in E\}$ to be the set of nodes j that are neighbors of node i (i.e., that join node i by an edge). Let x_{ij}^α denote the number of units of asset α transferred from node i to node j . In restricting consideration to admissible transfers, we assume each node i has an upper limit $U_i^{\alpha:R}$ on the number of units of any given asset $\alpha \in R_i$ that can be admissibly transferred from other nodes to node i and an upper limit $U_i^{\alpha:S}$ on the number of units of $\alpha \in S_i$ that can be transferred from i to other nodes. Formally, these conditions may be expressed as

$$\sum (x_{ji}^\alpha : j \in N_i) \leq U_i^{\alpha:R} \quad i \in N \text{ and } \alpha \in R_i \tag{1}$$

$$\sum (x_{ij}^\alpha : j \in N_i) \leq U_i^{\alpha:S} \quad i \in N \text{ and } \alpha \in S_i \tag{2}$$

We also impose an equation that requires the total number of assets transferred from a given node i to other nodes j to equal the total number of assets transferred in return from other nodes j to node i . Specifically, for a given node $i \in N$, we observe that the quantity $\sum (x_{ij}^\alpha : \alpha \in S_i \cap R_j \text{ and } j \in N_i)$ identifies the total number of units that can be admissibly transferred from node i to all nodes j and similarly, the quantity $\sum (x_{ji}^\alpha : \alpha \in S_j \cap R_i \text{ and } j \in N_i)$ identifies the total number of units that can be admissibly transferred from all nodes j to node i . We require these two quantities to be equal by stipulating

$$\sum (x_{ij}^\alpha : \alpha \in S_i \cap R_j \text{ and } j \in N_i) = \sum (x_{ji}^\alpha : \alpha \in S_j \cap R_i \text{ and } j \in N_i) \quad i \in N \tag{3}$$

Finally, we impose an additional limit U_i on the number of all assets α that can be admissibly transferred from node i to other nodes, expressed as

$$\sum (x_{ij}^\alpha : \alpha \in S_i \cap R_j \text{ and } j \in N_i) \leq U_i \quad i \in N \tag{4}$$

As a result of Eq. (3), this inequality is equivalent to

$$\sum (x_{ji}^\alpha : \alpha \in S_j \cap R_i \text{ and } j \in N_i) \leq U_i \quad i \in N \tag{4'}$$

Subject to these conditions, in problem AEP0 we seek to maximize the total number of admissible exchanges, hence yielding the formulation

$$\begin{aligned} &\text{Maximize } \sum (x_{ij}^\alpha : i \in N, j \in N_i, \alpha \in S_i \cap R_j) \\ &\text{subject to (1), (2), (3), (4) and} \\ &x_{ij}^\alpha \geq 0, \quad i \in N, j \in N_i, \alpha \in S_i \cap R_j \end{aligned} \tag{0}$$

We can also replace (0) by a variety of other objectives, such as

$$\text{Maximize } \sum (p_i^\alpha x_{ij}^\alpha : i \in N, j \in N_i, \alpha \in S_i \cap R_j) \tag{0'}$$

where p_i^α is a positive monetary value that node i attaches to receiving asset α from the set R_i .

We now take the step of transforming the foregoing preliminary AEP0 formulation into a network optimization formulation. Under the assumption that the data are integers, this allows us to automatically obtain solutions where the variables receive integer values when an extreme point network algorithm is used. More broadly, it gives a foundation for generating solutions to the AEP0 model by a corresponding

basic version of our combinatorial chaining approach. From this, we will be able to treat related more complex AEP models by natural extensions that combine the network optimization and combinatorial chaining components. The transformation of AEPO to a network formulation significantly increases the problem size, but offsets this by making the problem sparser, while our combinatorial chaining algorithm for this formulation is able to work with a memory based on the number of nodes rather than the number of arcs in the network, dramatically reducing both the amount of computation and the memory involved.

Transforming AEPO to a network formulation

The transformation of AEPO to an equivalent network formulation, which we call NetAEPO, arises by replacing the graph G by a graph $G^* = G^*(N^*, A^*)$ consisting of a set of nodes N^* and a set of arcs (directed edges) A^* as follows.

To emphasize the arc orientation in creating G^* , we find it useful to augment the customary representation of an arc from a node p to a node q as an ordered pair (p, q) by alternatively writing it in the form $p \rightarrow q$, which adds clarity when p and/or q is itself represented as an ordered pair. Lower bounds on all arc flows are assumed to be 0.

To generate G^* , we divide each node $i \in N$ into two nodes, $i[R]$ and $i[S]$, and create an arc $i[R] \rightarrow i[S]$. In addition, for each $i \in N$ and $\alpha \in R_i$ we create new nodes $(\alpha, i[R])$, producing $\sum(|R_i| : i \in N)$ nodes, and create arcs $(\alpha, i[R]) \rightarrow i[R]$ (from node $(\alpha, i[R])$ to node $i[R]$) which results in $\sum(|R_i| : i \in N)$ arcs (the same as the number of nodes $(\alpha, i[R])$). Similarly, for each $i \in N$ and $\alpha \in S_i$ we create new nodes $(\alpha, i[S])$, producing $\sum(|S_i| : i \in N)$ nodes, and create arcs $i[S] \rightarrow (\alpha, i[S])$, creating $\sum(|S_i| : i \in N)$ arcs (the same as the number of nodes $(\alpha, i[S])$). If $\alpha \in R_i$, we assume there is at least one neighbor of node i that can receive the asset α , or else we can drop α from R_i . Similarly, if $\alpha \in S_i$, we assume there is at least one neighbor of node i that can send α or else we can remove α from S_i . If, as a result of these removals, either R_i or S_i becomes empty, we can remove node i from N .

Finally, for each $i \in N$ and for each $j \in N_i$ and for each $\alpha \in S_i \cap R_j$, each node $(\alpha, i[S])$ joins by an arc $(\alpha, i[S]) \rightarrow (\alpha, j[R])$ to node $(\alpha, j[R])$. We call these the α -linking arcs of G^* , since the same asset α is referenced by both nodes of each of these arcs. The number of these arcs is $\sum(|S_i \cap R_j| : i \in N, j \in N_i)$.

From this construction we see that N^* consists of $2n + \sum(|R_i| + |S_i| : i \in N)$ nodes and A^* contains $n + \sum(|R_i| + |S_i| : i \in N) + \sum(|S_i \cap R_j| : i \in N, j \in N_i)$ arcs.

To create NetAEPO from G^* , we introduce flows on the arcs governed by bounds as follows. Each arc $i[R] \rightarrow i[S]$ receives an upper bound on its flow of U_i from (4). Correspondingly, each of the $(\alpha, i[R]) \rightarrow i[R]$ arcs receives an upper bound on its flow of $U_i^{\alpha:R}$ from (1) and each of the arcs $i[S] \rightarrow (\alpha, i[S])$ receives an upper bound on its flow of $U_i^{\alpha:S}$ from (2). Finally, the α -linking arcs of G^* are not given upper bounds (i.e., their upper bounds may be treated as infinity). All lower bounds are implicitly 0.

It is assumed that U_i satisfies $U_i \leq \text{Min}(\sum(U_i^{\alpha:R} : \alpha \in R_i), \sum(U_i^{\alpha:S} : \alpha \in S_i))$, that is, the upper bound U_i on the flow across arc $i[R] \rightarrow i[S]$ is limited by the smaller of the sum of upper bounds on the arcs $(\alpha, i[R]) \rightarrow i[R]$ entering $i[R]$ and the sum of upper bounds on the arcs $i[S] \rightarrow (\alpha, i[S])$ leaving $i[S]$. (Later we also describe variations in which we additionally introduce lower bounds $L_i^{\alpha:R}$ and/or $L_i^{\alpha:S}$ on the arcs $(\alpha, i[R]) \rightarrow i[R]$ and arcs $i[S] \rightarrow (\alpha, i[S])$.)

Because we start from the symmetric graph G in undirected edges to produce the graph G^* with directed arcs underlying NetAEP0, $j \in N_i$ implies $i \in N_j$. We additionally observe that no asset α is contained in both R_i and S_i for any given i , under the assumption that if node i sees a benefit in receiving a unit of $\alpha \in R_i$, then it will not be willing to relinquish a unit of α by including it in S_i . Exceptions can be imagined, as where i may be willing to give up a particular $\alpha' \in R_i$ if it is able to receive a more highly valued asset $\alpha'' \in R_i$. Such exceptions can be modeled by extensions of the constructions used here but make the formulation larger and more complex. Nevertheless, our basic algorithm can be modified to handle these and other variations without entailing the complexity introduced by an extended mathematical formulation.

The foregoing description of G^* and the conditions defining NetAEP0 can be translated into an algorithm for generating the network. As part of this we show how to attach numerical indexes denoted by $k = 1$ to n^* to the nodes in N^* so that NetAEP0 may be represented as a network in a standard format. We refer to lower bounds as well as upper bounds on arcs for generality, although in direct transformation of AEP0 to NetAEP0 the lower bounds will be 0.

Algorithm: Generate NetAEPO

```

for each  $i \in N$  do
  Create the nodes  $i[R]$  and  $i[S]$  and the arc  $i[R] \rightarrow i[S]$ , by assigning the index
   $k = i$  to the node  $i[R]$  and the indexes  $k = i + n$  to the node  $i[S]$ .
  Attach the lower and upper bounds  $L_i$  and  $U_i$  to the arc  $i[R] \rightarrow i[S](i \rightarrow i + n)$ .
end for
Set  $k = 2n$ 
for each  $i \in N$  do
  (Create the “S-labeled” asset node  $(\alpha, i[S])$  and associated arc  $i[S] \rightarrow (\alpha, i[S])$ 
  for each asset  $\alpha \in S_i$ .)
  for each  $\alpha \in S_i$  do
    Set  $k := k + 1$  and create the asset node  $(\alpha, i[S])$ , assigning it the index  $k$ .
    Create the arc  $i[S] \rightarrow (\alpha, i[S])$  and attach lower and upper bounds  $L_i^{\alpha:S}$  and
     $U_i^{\alpha:S}$ .
  end for
end for
for each  $j \in N$  do
  (Create the “R-labeled” asset node  $(\alpha, j[R])$  and associated arc
   $j[R] \rightarrow (\alpha, j[R])$  for each asset  $\alpha \in R_j$ .)
  for each  $\alpha \in R_j$  do
    Set  $k := k + 1$  and create the asset node  $(\alpha, j[R])$ , assigning it the index  $k$ .
    Create the arc  $(\alpha, j[R]) \rightarrow j[R]$  and attach lower and upper bounds  $L_j^{\alpha:R}$  and
     $U_j^{\alpha:R}$ .
  end for
end for
for each  $i \in N$  do
  (Create the “S to R” asset arcs  $(\alpha, i[S]) \rightarrow (\alpha, j[R])$  associated with  $i$  for each
   $\alpha \in S_i$ .)
  for each asset  $\alpha \in S_i$  do
    for each neighbor  $j \in N_i$  do
      if  $\alpha \in R_j$  then
        Create the asset arc  $(\alpha, i[S]) \rightarrow (\alpha, j[R])$  with no bounds (i.e., a lower
        bound of 0 and an upper bound of infinity).
      end if
    end for
  end for
end for

```

Costs or profits may be attached to the arcs of the network NetAEPO according to the objective that is desired to be achieved. Asset arcs, which are linking arcs, should be assigned a 0 cost or profit.

An illustration of the structure of NetAEPO is given in “Appendix 2”.

5 Basic version of combinatorial chaining

A classical theorem of network flows (Fulkerson and Ford 1962) implies that a feasible solution to NetAEPO can be decomposed into a sum of incidence vectors of cycles (not necessarily disjoint or uniquely determined). Such cycles are of interest for the

Asset Exchange Problem in both its simpler AEP0 form and its more complex forms because they identify a collection of participants who can enter into a succession of mutually beneficial asset exchanges. Such a collection is not unduly difficult to identify by reference to a solution to the NetAEP0 formulation but requires additional effort. Fulkerson and Ford's max flow algorithm would automatically identify (augmenting) paths from source to sink in the network, which has some similarity with combinatorial chaining. But a standard network flow algorithm for solving NetAEP0 is not capable of being directly adapted to provide good solutions to more complex variations of the AEP that abound in practical applications, thus motivating the creation of the adaptive combinatorial chaining approach.

Adopting the netform perspective (Glover et al. 1992), combinatorial chaining is designed both to exploit the structure of the basic AEP network formulation and to be susceptible to extensions for solving a variety of AEP variations found in practice. This harmonizes with the Quantum Bridge Analytics perspective where quantum computing can be applied to solve portfolio optimization problems expressed as QUBO models for individual investors or institutions, and more generally leads to consideration of a QUBO-Plus formulation of the third type. Combinatorial chaining can then be applied to the appropriate AEP variation to integrate and improve these individual solutions in a global strategy that accrues to the benefit of each participant.

The strategy underlying the basic form of combinatorial chaining operates by generating successions of directed trees (or arborescences in graph theory) rooted at different nodes. Conditions are monitored to disclose when a directed tree can be extended by connecting a tip of one of its branches to the root, thus creating a cycle that constitutes a mutually beneficial exchange. The process differs from classical tree generation algorithms by introducing multiple categories of tree predecessors and establishing a mechanism to trace the predecessors that differentiates between the categories effectively. This introduction of multiple categories of tree predecessors and mechanisms for tracing them likewise causes our method to operate differently from classical min cost flow algorithms based on generating augmented paths (Barr et al. 1978; Glover et al. 1986). This departure from classical approaches arises because many of the more general AEP models belong to the class of multi-commodity network flow problems (Hu 1963; Assad 1978), which are more complex than standard "pure" network flow problems, and normally cannot be transformed into a pure network problem as we have accomplished for AEP0. Rather than being a disadvantage, however, this complexity enables the chaining mechanism to be adapted to AEP variations beyond AEP0.

More broadly, the combinatorial chaining mechanism we employ is closely related to the ejection chain procedures for combinatorial optimization noted in Sect. 1. In its more advanced forms outlined in Sect. 6, it is additionally related to the path relinking approaches that are joined with ejection chains in Yagiura et al. (2006, 2007) and that produce the leading methods for the QUBO problem in Wang et al. (2012), Samorani et al. (2019), Glover et al. (2020, 2021).

Rudimentary combinatorial chaining for the NetAEP0 model

Combinatorial chaining for the basic NetAEP0 model makes use of arrays denoted $\text{FlowR}(\alpha, i[R])$ to record flows on the arcs $(\alpha, i[R]) \rightarrow i[R]$ and arrays denoted $\text{FlowS}(\alpha, i[S])$ to record the flows on the arcs $i[S] \rightarrow (\alpha, i[S])$. Hence, for each $i \in N$, we require $\text{FlowR}(\alpha, i[R]) \leq U_i^{\alpha:R}$ for each $\alpha \in R_i$, and require $\text{FlowS}(\alpha, i[S]) \leq U_i^{\alpha:S}$ for each $\alpha \in S_i$. Flows on the arcs $i[R] \rightarrow i[S]$ are recorded in an array $\text{Flow}(i)$ for each $i \in N$. All flow values are initialized to 0.

It is convenient to refer to the nodes $(\alpha, i[R])$, $(\alpha, i[S])$ and i (the latter collectively representing the two nodes $i[R]$ and $i[S]$) as *open* when their associated flows $\text{FlowR}(\alpha, i[R])$, $\text{FlowS}(\alpha, i[S])$ and $\text{Flow}(i)$ do not reach their upper bounds and *closed* otherwise. (A bit can be set for each such node to determine its open/closed status.)

We refer to two types of predecessor arrays $\text{PredR}(i)$ and $\text{PredS}(i)$, $i \in N$, accompanied by associated arrays $\text{AssetR}(i)$ and $\text{AssetS}(i)$ explained subsequently. The arrays $\text{PredR}(i)$ and $\text{PredS}(i)$ are initialized to 0 to indicate predecessors are not yet assigned.

The method performs forward scans and reverse scans to examine nodes $i \in N$ (and from there to examine the arcs these nodes can become linked to in a chain). When the tip of the tree can successfully be linked to the root, a *breakthrough* occurs by establishing the existence of an exchange cycle that is mutually beneficial for all its participants. Breakthrough is accompanied by appropriately updating (increasing) the flows on arcs of the cycle.

The basic version of the chaining algorithm only performs forward scans but gives the foundation for performing reverse scans as well, as subsequently described. We first explain the nature of the forward scan routine and then give a more formal description.

Rationale of the Forward Scan Routine:

The Forward Scan Routine is embedded in a Main Routine that maintains a set N^o identifying the open nodes, initialized by $N^o = N$. Nodes to be scanned are placed in a set denoted ScanSet that begins with a chosen node $i^* \in N^o$. During the Forward Scan Routine, ScanSet acquires other nodes $i \in N^o$ to form a tree that yields a collection of chains rooted at node i^* . The tree is generated by successively selecting new nodes i from ScanSet as long as $\text{ScanSet} \neq \emptyset$.

For each node i selected from ScanSet , consider each asset $\alpha \in S_i$; i.e., each asset α that node i is willing to send to another node. Given node i , additionally consider each neighbor j of i that contains $\alpha \in R_j$; i.e., each neighbor j that desires to receive α . (Formally, we refer to the set $\text{NR}_i^\alpha = \{j \in N_i : \alpha \in R_j\}$, which consists of those neighbors j of node i such that R_j contains α .) If node j is not already in the tree, i.e., if it has no predecessor (as indicated by $\text{PredS}(j) = 0$), then it can acceptably be added to the tree by adopting node i as its predecessor. For this, we set $\text{PredS}(j) = i$ together with $\text{AssetS}(j) = \alpha$, which records the fact that each chain in the tree that passes through this particular (i, j) link is accompanied by sending asset α from node i to node j .

If now $j = i^*$ (which can result because i^* is not assigned a predecessor initially), we have discovered a chain beginning with node i^* that results in a loop which qualifies as a mutually beneficial exchange cycle (where each participant receives a desired asset

and in return sends a willingly exchanged asset). The Breakthrough Routine handles this outcome by identifying the cycle and updating the flows and the structure of G^* appropriately.

Following the updates of the Breakthrough Routine, the scanning routine is reinitiated within the Main Routine by selecting a new i^* from N^o (where i^* may be the same as before if it is not removed from N^o during breakthrough).

Alternatively, the scan from a given node i^* may terminate with ScanSet empty and without achieving breakthrough. In this case, i^* is removed from N^o and once more the scanning routine is reinitiated within the Main Routine to select a new i^* from N^o .

We let $N_i^o = N_i \cap N^o$ denote the (current) neighbors of node i that are in N^o . Hence N_i^o , which starts the same as N_i , may shrink as nodes are removed from N^o . This also modifies the definition $NR_i^\alpha = \{j \in N_i : \alpha \in R_j\}$ to become $NR_i^\alpha = \{j \in N_i^o : \alpha \in R_j\}$, identifying the neighbors of i in N^o that desire to receive asset α .

Termination of the Main Routine occurs when N^o contains only a single node ($|N^o| = 1$), since then this node has no other nodes it can exchange with.

The formal design of the algorithm is as follows.

Algorithm: Basic Combinatorial Chaining

Initialization.

Set all flow values to 0. Initialize the set N^o of open nodes by setting $N^o = N$.

Main Routine

while $|N^o| > 1$ **do**

Set all predecessor arrays to 0.

Choose $i^* \in N^o$ and create ScanSet = $\{i^*\}$.

Execute the *Forward Scan Routine* (as follows)

while ScanSet $\neq \emptyset$ **do**

Select a node $i \in$ ScanSet

for each $\alpha \in S_i$ **do**

for each $j \in NR_i^\alpha (= \{j \in N_i^o : \alpha \in R_j\})$ **do**

if PredS(j) = 0 **then**

(j has not been visited before on a Forward Scan)

Set PredS(j) = i and AssetS(j) = α .

if $j = i^*$ **then**

Execute the *Breakthrough Routine* (below)

(Update flows and potentially remove nodes from N^o .)

Break (leave Forward Scan Routine to choose a new $i^* \in N^o$ in the Main Routine if $|N^o| > 1$).

end if

else

Let ScanSet := ScanSet $\cup \{j\}$.

end if

end for

end for

ScanSet := ScanSet $\setminus \{i\}$ (remove i from ScanSet)

(The scan of node i is complete.)

end while

(End of the Forward Scan Routine)

end while

(End of the Main Routine)

The algorithm can be modified to save part of the tree after the completion of each forward scan, but the computational savings will not usually be enough to warrant the effort. Reverse scanning provides a more interesting modification and can be accomplished by interchanging R and S in each of the instructions of the Forward Scanning Routine. Forward scanning and reverse scanning can also be done together, switching from one to the other on selected iterations. In this case, breakthrough is recognized when $j = \text{PredS}(i)$ on a forward scan yields $\text{PredR}(j) > 0$ (where $\text{PredR}(j)$ was set on a reverse scan), or when $j = \text{PredR}(i)$ on a reverse scan yields $\text{PredS}(j) > 0$ (where $\text{PredS}(j)$ was set on a forward scan). To show how reverse scanning can be joined with forward scanning, “Appendix 3” gives an example where a single iteration of reverse scanning is applied before launching the forward scanning algorithm.

The Breakthrough Routine that accompanies the Forward Scanning Routine may now be described as follows. The preceding observations and the example in “Appendix 3” disclose how to modify this routine for reverse scanning or for combinations of forward and reverse scanning.

Algorithm: Breakthrough Routine

Compute the maximum feasible flow increment ΔFlow on the augmenting cycle

$\Delta\text{Flow} = \text{Big}$ (a large positive number)

$i = i^*$

Stop = False

while Stop = False **do**

$\alpha = \text{AssetS}(i)$

$\Delta R = U_i^{\alpha:R} - \text{FlowR}(\alpha, i[R])$

$i = \text{PredS}(i)$

$\Delta S = U_i^{\alpha:S} - \text{FlowS}(\alpha, i[S])$

$\Delta i = U_i - \text{Flow}(i)$

$\Delta\text{Flow} = \text{Min}(\Delta R, \Delta S, \Delta i, \Delta\text{Flow})$

if $i = i^*$ **then**

 Stop = True

end if

end while

Update flows and remove nodes associated with saturated arcs

$i = i^*$

Stop = False

while Stop = False **do**

$\alpha = \text{AssetS}(i)$

$\text{FlowR}(\alpha, i[R]) = \text{FlowR}(\alpha, i[R]) - \Delta\text{Flow}$

if $\text{FlowR}(\alpha, i[R]) = U_i^{\alpha:R}$ **then**

 close arc $(\alpha, i[R])$ by setting $R_i := R_i \setminus \{\alpha\}$ (removing α from R_i)

end if

$i = \text{PredS}(i)$

$\text{FlowS}(\alpha, i[S]) = \text{FlowS}(\alpha, i[S]) - \Delta\text{Flow}$

if $\text{FlowS}(\alpha, i[S]) = U_i^{\alpha:S}$ **then**

 close arc $(\alpha, i[S])$ by setting $S_i := S_i \setminus \{\alpha\}$ (removing α from S_i)

end if

$\text{Flow}(i) = \text{Flow}(i) - \Delta\text{Flow}$

if $\text{Flow}(i) = U_i$ **then**

 close arc $(i[R], i[S])$ setting $N^o := N^o \setminus \{i\}$

end if

if $i = i^*$ **then**

 Stop = True

end if

end while

6 Advanced forms of combinatorial chaining for more complex AEP models

There are problems that are too complex to be given mathematical formulations that fully capture their subtleties and that are simultaneously capable of being solved by standard math programming algorithms. In adopting the perspective of Quantum Bridge Analytics, we embrace strategies for such problems that allow their objectives to be pursued approximately and flexibly, thus admitting approaches that solve variations

of these problems to emphasize alternative problem components in an adaptive fashion. As we have emphasized, our basic combinatorial chaining procedure allows this to be done when joined with network optimization by giving advanced methods that yield access to more complex AEP variants.

We show how this can be achieved for two chief extensions of the basic formulation that encompass a broad range of applications. The associated modified versions of combinatorial chaining provide flexible approximation methods that can be embedded in metaheuristic algorithms and afford the possibility of being incorporated into hybrid classical/quantum systems. In common with the most effective algorithms for QUBO problems, a natural basis for these extensions derives from adaptive memory strategies such as embodied in tabu search and path relinking (Wang et al. 2012; Samorani et al. 2019; Glover et al. 2020, 2021).

Prioritizing the assets exchanged

In some applications of the AEP, participants may wish to prioritize certain exchanges of assets over others, preferring more strongly to receive particular assets and being more willing to relinquish certain other assets. Priorities attached to these preferences may also differ among different participants. Upon assigning numerical values to capture these preferences (as by indicating a dollar amount that different individuals attach to the value of different exchanges, or by making recourse to an agreed-upon set of subjective weights), the combinatorial chaining algorithm can be extended by prioritizing the selection of the elements i^* in N^o or the choice of elements i in ScanSet, in each instance selecting the highest priority element from those available.

Priorities can also be used by such an extension to improve the choices for participants whose exchanges were less favorable on previous executions of the algorithm, since an effort to achieve a best overall collection of exchanges (such as a maximum number of beneficial exchanges) can result in better outcomes for some participants than for others. This means of exploiting the freedom to choose different elements in executing the basic steps of combinatorial chaining yields an approximation method for a problem whose subtleties render it unsuitable for a classical mathematical formulation, while allowing the flexibility to be adapted to different types of priorities. Such priorities can be introduced in the network formulation and embodied in probabilities for selecting moves in metaheuristic adaptations as in probabilistic tabu search (Xu et al. 1997; Guemri et al. 2019). Combinatorial chaining provides the underlying structure for guiding the search to produce feasible solutions.

Priorities can also be employed to create larger breakthroughs earlier in the process of generating combinatorial chains, as by giving higher priority to participants with larger capacities (upper bounds) on the flows they can receive. The priorities can be based on measures applied to each base node (participant), such as total sums of capacities or means of capacities adjusted by standard deviations, and so forth. Refinements arise by considering the priorities of neighbors. For example, a new priority can be created for a node that is a weighted combination of its current priority and the priorities of neighbor nodes, where weights for neighbors are less than for the node under consideration. Such a process may also be repeated, using the new priorities

as a basis for constructing another round of new priorities. (Additional repetitions may be expected to yield progressively less advantage.)

Particular applications give their own criteria for determining priorities. In exchanges of cryptocurrencies, for example, larger investors face the most negative impact by failing to make exchanges of a size deemed satisfactory, so assigning higher priorities to exchanges of such investors will usually result in the highest increase in utility. Using such priorities, choosing a node i^* from N^o with the highest priority to become the root of the current directed tree, followed by choosing highest priority nodes i from ScanSet to continue building the tree, provides a compelling and easily implemented strategy.

As previously observed, there may also be situations where it can be relevant to place lower bounds as well as upper bounds on the number of units of different assets exchanged by different participants. In a cryptocurrency application, for example, an investor may only be interested in transactions that result in receiving a specified number of units of a given asset. To illustrate, an investor represented by a node i may seek an exchange in which i receives precisely 100 units of Ethereum (ETH), represented by asset $\alpha (\in R_i)$ (accompanied, for example, by i sending units of Bitcoin (BTC) or Lumen (XLM) to other nodes). The AEP network model then captures this by putting a lower bound of 100 and an upper bound of 100 on the ETH arc $(\alpha, i[R]) \rightarrow i[R]$, giving $L_i^{\alpha:R} = U_i^{\alpha:R} = 100$. The situation where an investor may have an exact demand for an asset (modeled by setting the lower bound equal to the upper bound), and where this demand cannot be satisfied by an exchange involving any single other investor, is sometimes called splitting, i.e., the demand must be split into different transactions with different investors. Combinatorial chaining automatically handles splitting situations as well as other much more general situations. A simple illustration is where investor i will only consider an exchange that brings in at least $L_i^{\alpha:R} = 50$ units of ETH, but would prefer to receive more units, up to a limit of $U_i^{\alpha:R} = 100$. Any number of other investors, some who may not be neighbors of i , may be involved in transactions identified by combinatorial chaining.

In cases like these where the AEP model includes lower bounds on numbers of units received, exchanges can be prioritized in two phases, where Phase 1 is devoted to satisfying as many of the lower bounds as possible, and Phase 2 then sends additional flow through the network subject to satisfying upper bounds. These two phases are not required to have the same priorities for selecting nodes on exchange cycles.

Machine learning provides a natural way to facilitate priority generation. A strategy of varying the priorities may yield better overall outcomes for a particular objective, for example, and machine learning can be used to help identify a strategy that leads to the most desirable results. An instance of machine learning called Programming by Optimization (Hoos 2012) is often effective for choosing parameters for optimization algorithms and may be useful in determining priorities in the combinatorial chaining context. Learning can also be employed as clustering-based metaheuristics (Samorani et al. 2019).

Generalized networks

An important variation of the AEP arises where a unit of one asset may be exchanged for more or less than one unit of another asset. Networks in which the number of units received at the destination node (to-node) of an arc may differ from the number of units sent from the origin node (from-node) of an arc are called *generalized networks* (Glover et al. 1990, 1992), and the factor that determines the difference between the units sent and received is called the *arc multiplier*. For example, an arc multiplier of 1.5 implies that the to-node receives 1.5 units for every unit sent from the from-node. A variety of situations exist where assets may be exchanged other than on a one-to-one basis.

A convenient feature of the basic combinatorial chaining algorithm is that such multiplier effects can be captured by joining the treatment of priorities with a modification of the Breakthrough Routine. The amount of flow transmitted across a chain of generalized arcs from the root node i^* to a subsequent node i equals the product of the multipliers on the arcs between i^* and i . Thus, for example, if the chain consists of the succession of arcs (i^*, i^1) , (i^1, i^2) , (i^2, i^3) , with $i^3 = i$, and if the multipliers on these three arcs are 0.6, 2.0 and 1.2, then a unit of flow sent from node i^* becomes $0.6 \times 2.0 \times 1.2 = 1.44$ units of flow received at node i^3 . The Breakthrough Routine can be readily modified to incorporate this effect, using it to identify the limits on flows required to compute updated flows across the entire cycle and to determine which assets or elements must be removed from their associated sets due to these updates.

The approaches of introducing exchange priorities and capitalizing on the ability to incorporate arc multipliers in association with generalized networks can be combined to cover an additionally expanded range of practical problems, which may be usefully exploited by metaheuristic algorithms in the QBA context.

7 Concluding remarks

The relevance of Quantum Bridge Analytics for real world applications has been demonstrated by showing an important instance where we are able to apply the QBA perspective to the challenging Asset Exchange Problem, which opens up numerous applications in financial investment, resource allocation, economic distribution and collaborative decision making. The linkage of network optimization with metaheuristic optimization via combinatorial chaining gives rise to an Asset Exchange Technology that can address and solve a wide range of practical variations.

Present day quantum computers can only handle small AEP problems, due to the limited number of qubits they encompass, but the integration of network formulations and combinatorial chaining is capable of accommodating AEP problems of significantly greater dimension. Through these connections, the AEP model gives an important class of optimization problems that can be usefully approached within the QBA domain, providing a foundation for further advances in the future as quantum computing technology becomes more mature.

Acknowledgements The authors are indebted to Yves Crama for insightful observations that have improved the quality of this paper.

Compliance of ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Appendix 1: Blockchains and decentralized market making

Decentralized market making is an intriguing concept that would require a detailed exploration, as it will likely emerge as critical factor for enabling scalable liquidity. But there are many questions to be answered. For example, what is the value of contributions by the decentralized market makers? Also, could these small investments held by the market—provided to equalize values in an exchange—be aggregated into baskets, and could those baskets be traded? How do we accurately assess the risks of items in baskets, to flow them up to the basket, to avoid “toxic assets” being included?

Finally, it should be noted that a computational system or agent that learns what a user wants to buy or sell, or might be willing to trade, would be quite valuable as an e-commerce tool because it provides a means to unveil the deeper purchase intentions of users. AI based agents could assist not only in the process of helping the user to determine what they might be willing to trade for or buy but could even help the user discover new purchase intentions that might lead to greater personal satisfaction. In other words, instead of just contributing to the accumulation of more useless stuff in their lives, such a system could explore more complex human values, as opposed to those reflecting desires and whims stimulated by media and advertising.

For example, if an AI held a model that understood the OCEAN Big Five personality traits, which was used so effectively by Cambridge Analytica in 2016, it could predict that the user has a high degree in a single trait, openness to new experiences. By balancing knowledge around both investment planning and personality traits, the advisor could provide more balanced advice to the user that would lead to greater personal satisfaction and fulfillment. A strictly financial based AI advisor would simply recommend one asset class over other, or the diversification into additional classes. But an AI advisor that used both financial optimization as well as heuristics about human personality and psyche, understanding the complex needs of the investor, might suggest to keep 95% of the portfolio within financial instruments, but propose that 5% could be invested in experiential learning for the user, in other words, investing in him or herself. This could include travel to learn a new language or a workshop to learn a new skill, possibly with permission to tap into the user’s online “bucket list” - the list of things you’d like to do before you “kick the bucket.”

To put this into the context of the AEP problem and combinatorial chaining, consider a situation with User A who has inherited a somewhat odd abstract painting from a distant relative in France, that doesn’t have much value on the resale market in America. However, on a combinatorial exchange market, there may be a chance of trading it for something not only less objectionable but desirable for all parties. Her asking price is a value of \$3,000. Now, because her interaction with the exchange is managed by

a user agent with access to her private “bucket list,” the trusted agent can now look for something that matches items on his list. It turns out that she has always wanted to take a class at the Cordon Bleu cooking school and to learn some French. So our agent can scan against other agents and listings, to find User B who wants to trade a \$3000 workshop pass at Cordon Bleu for ten day stay in a beachfront Airbnb on some nice tropical island. The combinatorial chain holds that in place while finding a third or fourth transaction to make the combinatorial exchange pareto-optimal for all users. Fortunately, it finds User C who has a modest bungalow on a beach in the Marquesas, which doesn’t get much Airbnb interest because it is too remote. However, that person looks at the painting, and realizes it was painted by the singer Jacques Brel, who was a great singer but lousy painter, and actually has quite a bit of value in the Marquesas because Jacques Brel spent his last days on the island of Hiva Oa, following the footsteps of Paul Gauguin and learning how to paint untamed landscapes that were so bad they looked abstract. So his agent offers a 3 week stay for that painting!

In this way, an AI-based financial advisor would advise in a more human and humane way. Thus, metaheuristic optimization via asset exchange technology could be applied directly to the issues of happiness, life goals and meaning. For user A, the lifelong goal of learning how to master the art of French cooking. For User B, a desperately needed vacation he couldn’t afford otherwise. And for User C, the lifelong goal of appearing on Antique Roadshow, to show off a barn find of a lifetime. We thus can ascend from cold process of optimizing utility functions to optimizing the human condition.

Appendix 2: Illustration of network structure of NetAEPO

The structure of the network NetAEPO created in Sect. 4 is illustrated in the following diagram, where the i nodes are represented in their duplicated form $i[R]$ and $i[S]$, giving rise to the arc $i[R] \rightarrow i[S]$, for a network with $N = \{1, \dots, 6\}$. The assets α are represented by the letters A, B, C, D and E, giving rise to asset nodes of the form $(\alpha, i[S])$ and $(\alpha, j[R])$ which are joined by arcs $(\alpha, i[S]) \rightarrow (\alpha, j[R])$ (called α -linking arcs in Sect. 4), where i and j may vary but the asset $\alpha (= A, B, \dots, \text{etc.})$ must be the same in each such arc. It should be noted that these linking arcs do not have limiting bounds on their flows other than an implicit lower bound of 0.

The arcs of the network are a succession of nodes that can be written in columns of R-labeled nodes and S-labeled nodes that follow a pattern that begins with the R-labeled i nodes $i[R]$, followed by the S-labeled i nodes $i[S]$, followed in turn by the S-labeled asset nodes $(\alpha, i[S])$, then followed by the R-labeled asset nodes $(\alpha, j[R])$ and finally followed by the R-labeled i nodes $i[R]$ to repeat the pattern. A further interesting pattern seen in the diagram is that all S-labeled nodes have exactly 1 arc entering but may have multiple arcs leaving, while all R-labeled nodes have exactly 1 arc leaving but may have multiple arcs entering. The i nodes are enclosed in circles in the diagram and the asset nodes are enclosed in rectangles.

Since the asset arcs (linking arcs) do not have bounds on their flows, the foregoing patterns imply that an asset arc whose S-labeled node has a single arc out can be collapsed to be represented only by the R-labeled node, and an asset arc whose R-

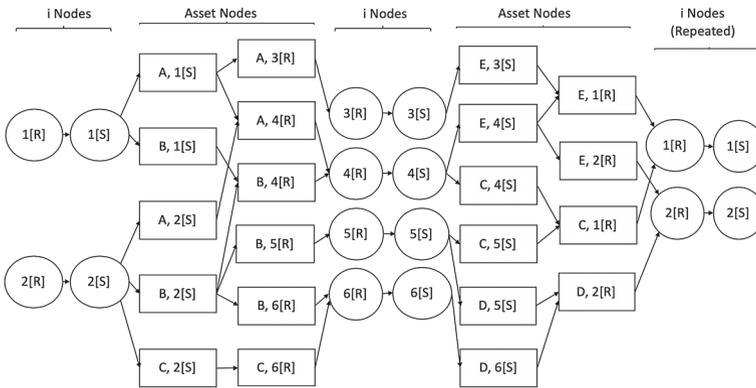


Fig. 1 Network structure

labeled node has a single arc in can be collapsed to be represented only by the S-labeled node. It should be emphasized that the staged structure shown in the diagram above is slightly misleading, since cycles typically vary in length and, in addition, duplicated i nodes may be encountered at various stages without implying they form a cycle that can be traced back to a previous instance of a duplicated node. The i indexes and the assets in the diagram have been ordered to show the patterns produced by arranging the nodes in columns. By contrast, the algorithm given in Sect. 4 for generating the network applies for any ordering of the indexes i in N and is independent of any ordering of the assets, which shows that such orderings are irrelevant in the general case.

Appendix 3: Illustration for reverse scanning

Algorithm: Basic Combinatorial Chaining Algorithm with a Reverse Scanning Step

Initialization.

Set all flow values and all predecessor arrays to 0. Initialize the set N^o of open nodes by setting $N^o = N$.

Main Routine

while $|N^o| > 1$ **do**

Set all predecessor arrays to 0.

Choose $i^* \in N^o$ and create $\text{ScanSet} = \{i^*\}$.

Execute the *Reverse Scan Routine* (as follows)
(recorded by setting $\text{PredR}(j) = i^*$).

if no Fertile nodes are found ($\text{Find} = \text{False}$) **then**

set $N^o := N^o \setminus \{i^*\}$

Continue the next iteration of the Main Routine (returning to choose a new $i^* \in N^o$ (accompanied by $\text{ScanSet} = \{i^*\}$) if $|N^o| > 1$).

end if

Execute the *Forward Scan Routine* (as follows)

while $\text{ScanSet} \neq \emptyset$ **do**

Select a node $i \in \text{ScanSet}$

for each $\alpha \in S_i$ **do**

for each $j \in \text{NR}_i^\alpha (= \{j \in N_i^o : \alpha \in R_j\})$ **do**

if $\text{PredS}(j) = 0$ **then**

(j has not been visited before on a Forward Scan)

Set $\text{PredS}(j) = i$ and $\text{AssetS}(j) = \alpha$.

if $\text{PredR}(j) > 0$ (node j is a Fertile node) **then**

Execute the *Breakthrough Routine* (below)

(Update flows and potentially remove nodes from N^o .)

Break (leave Forward Scan Routine to choose a new $i^* \in N^o$ in the Main Routine if $|N^o| > 1$).

end if

else

Let $\text{ScanSet} := \text{ScanSet} \cup \{j\}$.

end if

end for

end for

end while

(End of the Forward Scan Routine)

end while

(End of the Main Routine)

Algorithm: Reverse Scan Routine

```

Set Find = False
for each  $\alpha \in R_i$  do
  for each  $j \in \text{NS}_{i^*}^\alpha (= \{j \in N_{i^*}^o : \alpha \in S_j\})$  do
    if  $\text{PredR}(j) = 0$  then
      ( $j$  has not been visited before on a Reverse Scan)
      Set  $\text{PredR}(j) = i^*$  and  $\text{AssetR}(j) = \alpha$  and  $\text{Find} = \text{True}$ .
    end if
  end for
end for
if  $\text{Find} = \text{False}$  then
  no fertile nodes are discovered.
end if

```

Note: $\text{Find} = \text{False}$ at the end only if $\text{NS}_{i^*}^\alpha = \emptyset$ for all $\alpha \in R_{i^*}$. The check for $\text{PredR}(j)$ can be ignored and the assignments $\text{PredR}(j) = i^*$ and $\text{AssetR}(j) = \alpha$ and $\text{Find} = \text{True}$ can be executed for each j encountered. It doesn't matter that these assignments write over previous assignments in this case.

References

- Assad AA (1978) Multicommodity network flows—a survey. *Networks* 8(1):37–91
- Barr R, Elam J, Glover F, Klingman D (1978) A network augmenting path basis algorithm for transshipment problems. In: Fiacco A, Kortanek K (eds) *Extremal methods and systems analysis*. Springer, Berlin, pp 250–274
- Du Y, Kochenberger G, Glover F, Lu ZP, Liu DH, Hulandageri A (2020) Optimal solutions to the minimum sum coloring problem: a comparison of alternative models. Working paper, University of Colorado Denver
- Fitzpatrick L (2019) A complete beginner's guide to atomic swaps. *Forbes*, Sept 2, 2019 [https://www.forbes.com](https://www.forbes.com/sites/lukefitzpatrick) > sites > lukefitzpatrick
- Fulkerson DR, Ford LR Jr (1962) *Flows in networks*. Rand Corporation, Santa Monica
- Glover F (1996) Ejection chains, reference structures and alternating path methods for traveling salesman problems. *Discrete Appl Math* 65:223–253
- Glover F, Glover R, Klingman D (1986) The threshold assignment algorithm. In: Gallo G, Sandi C (eds) *Netflow at Pisa*, vol 26. *Mathematical programming study*. Springer, Berlin, pp 12–37
- Glover F, Phillips N, Klingman D (1990) Netform modeling and applications. Special issue on the practice of mathematical programming. *Interfaces* 20(1):7–27
- Glover F, Klingman D, Phillips N (1992) *Network models in optimization and their applications in practice*. Reviewed in *interfaces*. Wiley Interscience. Wiley, Hoboken, p 284
- Glover F, Kochenberger G, Du Y (2019) Quantum bridge analytics I: a tutorial on formulating and using QUBO models. *4OR Q J Oper Res Invited Surv* 17:335–371
- Glover F, Kochenberger G, Du Y, Hennig R, Wang H, Mniszewski S, Hulandageri A (2020) New advances for solving important classes of combinatorial optimization problems. In: 12th INFORMS conference on information systems and technology (CIST), virtual 2020 INFORMS annual meeting, November 7–13
- Glover F, Kochenberger G, Du Y (2021) Applications of the QUBO Model. In: Punnen AP (ed) *The quadratic unconstrained binary optimization problem*. Springer, Berlin
- Guemri O, Nduwayoya P, Todosijejevic R, Hanafi S, Glover F (2019) Probabilistic tabu search for the cross-docking assignment problem. *Eur J Oper Res* 277(3):875–885
- Helsgaun K (2000) An effective implementation of the Lin–Kernighan traveling salesman heuristic. *Eur J Oper Res* 126(1):106–130

- Helsgaun K (2009) General k-opt submoves for the Lin–Kernighan TSP heuristic. *Math Program Comput* 1(2–3):119–163
- Hoos H (2012) Programming by optimization. *Commun ACM* 55(2):70–80
- Hu TC (1963) Multi-commodity network flows. *Oper Res*. <https://doi.org/10.1287/opre.11.3.344>
- Kochenberger G, Ma M (2019) Quantum computing applications of QUBO models to portfolio optimization. White paper. University of Colorado, Denver, September 2019
- Kubicka E, Schwenk AJ (1989) An introduction to chromatic sums. In: CSC'89: proceedings of the 17th conference on ACM annual computer science conference, New York, NY, USA, ACM, pp 39–45
- Müller JC, Pokutta S, Martin A, Pape S, Peter A, Winter T (2017) Pricing and clearing combinatorial markets with singleton and swap orders. *Math Methods Oper Res* 85:155–177
- National Academies (2019) The national academies of sciences, engineering and medicine, 2019, quantum computing: progress and prospects. The National Academies Press, NY <https://doi.org/10.17226/25196>
- Nolan T (2013) Alt chains and atomic transfers. *Bitcointalk* <https://bitcointalk.org/index.php?topic=193281.0>
- Rego C, Glover F (2006) Ejection chain and filter-and-fan methods in combinatorial optimization. *4OR Q J Oper Res* 4(4):263–296
- Rego C, Gamboa D, Glover F (2016) Doubly-rooted stem-and-cycle ejection chain algorithm for the asymmetric traveling salesman problem. Special issue on metaheuristics in network optimization. *Networks* 68(1):23–33
- Samorani M, Wang Y, Lu Z, Glover F (2019) Clustering-driven evolutionary algorithms: an application of path relinking to the QUBO problem. In: Glover F, Samorani M (eds) Special issue on intensification, diversification and learning of the *Journal of Heuristics*, 25(4–5), 629–642
- Wang Y, Lu Z, Glover F, Hao J-K (2012) Path relinking for unconstrained binary quadratic programming. *Eur J Oper Res* 223(3):595–604
- Winter T, Rudel M, Lalla H, Brendgen S, Geißler B, Martin A, Morsi A (2011) System and method for performing an opening auction of a derivative. <https://www.google.de/patents/US20110119170>. Pub. No.: US 2011/0119170 A1. US Patent App. 12/618,410
- Xu J, Chiu S, Glover F (1997) Tabu search for dynamic routing communications network design. *Telecommun Syst* 8:1–23
- Yagiura M, Ibaraki T, Glover F (2006) A path relinking approach with ejection chains for the generalized assignment problem. *Eur J Oper Res* 169:548–569
- Yagiura M, Komiya A, Kojima K, Nonobe K, Nagamochi H, Ibaraki T, Glover F (2007) A path relinking approach for the multi-resource generalized quadratic assignment problem, vol 4638. *Lecture notes in computer science*. Springer, Berlin, pp 121–135

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Affiliations

Fred Glover¹  · Gary Kochenberger² · Moses Ma³ · Yu Du²

✉ Fred Glover
fred.glover@colorado.edu

Gary Kochenberger
gary.kochenberger@ucdenver.edu

Moses Ma
moses.ma@futurelabconsulting.com

Yu Du
yu.du@ucdenver.edu

- ¹ ECEE, College of Engineering and Applied Science, University of Colorado, Boulder, CO 80302, USA
- ² Business School, University of Colorado at Denver, Denver, CO 80217, USA
- ³ FutureLab Consulting, Mill Valley, CA 94941, USA