Received January 18, 2020, accepted January 30, 2020, date of publication February 12, 2020, date of current version March 11, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2973607

An Efficient, Scalable, and Robust Neuro-Processor-Based Concept for Solving Single-Cycle Traveling Salesman Problems in Complex and Dynamically Reconfigurable Graph Networks

JEAN CHAMBERLAIN CHEDJOU[1], KYANDOGHERE KYAMAKYA[1], AND NKIEDIEL ALAIN AKWIR[2]

[1]Institute of Smart Systems Technologies, Alpen-Adria University of Klagenfurt, 9020 Klagenfurt, Austria
[2]Department of Electrical Engineering, Université Libre des Pays des grands Lacs, Goma 09101, Democratic Republic of the Congo

Corresponding author: Kyandoghere Kyamakya (kyandoghere.kyamakya@aau.at)

This work was supported by the University of Klagenfurt (Austria).

ABSTRACT We develop, for the first time, and validate through some illustrative examples a new neuro-processor based concept for solving (single-vehicle) traveling salesman problems (TSP) in complex and dynamically reconfigurable graph networks. Compared to existing/competing methods for solving TSP, the new concept is accurate, robust, and scalable. Also, the new concept guarantees the optimality of the TSP solution and ensures subtours avoidance and thus an always-convergence to a single-cycle TSP solution. These key characteristics of the new concept are not always satisfactorily addressed by the existing methods for solving TSP. Therefore, the main contribution of this paper is to develop a systematic analytical framework to model (from a nonlinear dynamical perspective) the TSP, avoid/eliminate subtours, and guarantee/ensure convergence to the true/exact TSP solution. Using the stability analysis (nonlinear dynamics), analytical conditions are obtained to guarantee both robustness and convergence of the neuro-processor. Besides, a bifurcation analysis is carried out to obtain ranges (or windows) of parameters under which the neuro-processor guarantees both TSP solution's optimality and convergence to a single-cycle TSP solution. In order to validate the new neuro-processor based concept developed, two recently published application examples are considered for both benchmarking and validation as they are solved by using the developed neuro-processor.

INDEX TERMS Basic differential multiplier method, bifurcation analysis, constrained optimization, convergence to a single-cycle TSP tour/solution, dynamically externally reconfigurable TSP, local stability, neuro-processor, nonlinear optimization, subtours elimination, traveling salesman problem, validation through a benchmarking.

NOMENCLATURE

$\vec{x}$	Vector of decision variables
$\vec{\lambda}, \vec{\varphi}, \vec{\gamma}, \vec{\gamma}^{*}, \vec{\omega}, \vec{\mu}_l$	Vectors of multiplier variables
$\vec{x}, \vec{\lambda}, \vec{\varphi}, \vec{\gamma}, \vec{\gamma}^{*}, \vec{\omega}, \vec{\mu}_l$	Dependent variables (on t)
t	Independent variable
$f(\vec{x})$	Objective function
$g_{1j}, g_{2j}, g_{3i}, g_{3i}^{*}, g_{4i}, g_{5k,l}$	Constraints functions
$\tilde{L}\left(\vec{x}, \vec{\lambda}, \vec{\varphi}, \vec{\gamma}, \vec{\gamma}^{*}, \vec{\omega}, \vec{\mu}_l\right)$	Lagrange function
α	Step size of decision variables
$\beta_1, \beta_2, \beta_3, \beta_3^{*}, \beta_4, \beta_{5,l}$	Step sizes of multiplier variables
$\alpha, \beta_1, \beta_2, \beta_3, \beta_3^{*}, \beta_4, \beta_{5,l}$	Learning rate parameters
U	Vector of the costs of edges
A	Incidence matrix
D	Matrix of parallel edges

The associate editor coordinating the review of this manuscript and approving it for publication was Choon Ki Ahn.

S	Matrix of subtours
A^T, D^T, S^T	Transpose of A, D, S
A^+, A^{+T}	Absolute value of A, A^T
S_k	Set of subtours in a specific group of subtours
$\aleph$	Number of subtours in the set S_k
$\mathbb{N}$	Number of edges in a given subtour
N	Size of graph G
M	Magnitude of graph G
n	Index of a specific group of subtours
$d()/dt$	First order total derivative with respect to t
$\partial_x = \partial()/\partial x$	First order partial derivative with respect to x
"$\times$" and "$\circ$"	Product and elementwise product.

I. INTRODUCTION

A. POTENTIAL APPLICATIONS OF TSP IN SCIENCE AND ENGINEERING

The traveling salesman problem (TSP) is a traditional combinatorial optimization problem, which has been widely studied over 50 years in the fields of mathematics, artificial intelligence and computer science. The problem is defined as follows: ''given a network of n cities, the salesman has to travel to each city exactly once and return to the starting city with minimum utilization of resources" [1] (in general, this may be the shortest distance or in other words a ''spanning'' shortest path starting and ending at the starting city).

In intelligent transportation systems (ITS), for example, TSP is encountered in vehicle routing problems [2] such as pick-up and delivery problems (PDPs) [3], vehicle and crew scheduling [4], etc. Further interesting fields of application of TSP are: printed circuit boards manufacturing [5], data transmission in computer networks [6], power distribution networks [7], image processing and pattern recognition [8], robot navigation [9] and data partitioning [10], logistics and unmanned aircraft mission planning [11], just to name a few. The TSP, as a typical example of a canonical vehicle routing problem, has received tremendous attention during the past decade [12]–[18]. The TSP has also been applied in the assignment and scheduling strategies/procedures for homogenous teams of unmanned vehicles [19] and in static strategies for heterogeneous teams in multi-vehicle problems [20]. Further concepts of relevance for TSP have been developed such as the following ones: (a) time-dependent TSP [21], (b) a one-machine scheduling problem [22], (c) shorter tours detection problems based on real-time traffic data [23], (d) tolerances' origin problems and sensitivity analysis based on input data [24], just to name a few.

As a generalized form of TSP – which is however not considered in this paper, the multiple TSP [25] denoted by MTSP/a (a TSP solution with many cycles, many salesmen, and one base city/node) compared to the TSP (a solution with a single cycle), is a well-known NP-hard combinatorial optimization problem. A particular case of MTSP/b (with many base cities and one salesman) can be solved by the neuro-processor developed in this paper by disabling of the constraint related to the avoidance of subtours in the TSP solution (this can be achieved by setting to zero the coefficients related to that specific constraint).

B. TRADITIONAL TSP SOLVER CONCEPTS: CHALLENGES AND RELATED PROS AND CONS

A series of commonly used criteria for judging all relevant concepts from literature can be summarized as follows: (a) sure detection of the exact TSP solution (several TSP algorithms converge randomly between exact solutions and approximate solutions); (b) complexity or computational effort (several TSP algorithms are very time consuming); (c) scalability (several TSP algorithms do not converge in case of huge-size graphs or the computational effort raises exponentially with regards to graph size); (d) the guaranty for single- cycle TSP solution (several algorithms converge randomly between solutions with one-cycle (TSP) and solutions with multiple cycles (MTSP/b)); (e) good integration of the dynamic change of graph parameters such as weights values, size of the graph, magnitude of the graph, etc. (several TSP algorithms do not efficiently handle the dynamic change of graph parameters).

To date, several methods have been proposed for solving TSP, which are based on deterministic or probabilistic heuristics such as traditional graph search methods [27], simulated annealing (SA) [28], artificial neural networks (ANN) [29], genetic algorithms (GA) [30], ant colony optimization (ACO) [31], and particle swarm optimization (PSO) [32], just to name a few.

In essence, various evolutionary algorithms (EA) are inspired from the genetic and natural selection. EA can be successfully applied for solving difficult/complex traveling salesman problems (TSP). EA have been however frequently found inefficient in solving especially large TSP cases [14]. J. Holland presented genetic algorithms (GA) in 1970 to study the self-adaptation behavior of natural systems [37]. It is a random search algorithm using a roulette wheel selection to select chromosomes and it is also a global search algorithm. Nevertheless, GA has both strong tolerance and strong robustness, which are important features for solving complex combinatorial optimization cases. GA is too dependent on the initial population, is subject to premature convergence and shows poor local search capabilities. Many heuristic and metaheuristic supplements have been used to improve the performance of GA, but GA cannot achieve optimum solutions in finite time for graphs of high magnitude [38]. GA has the potential of being combined with other algorithms to perform the search for optimum solutions [15], [26]. Ant colony optimization (ACO) always looks for optimum solutions but requires more time for convergence and at times it remains stagnating for higher number of nodes/cities. A parallel implementation of ACO displays faster search of optimum solutions [39] yet remains inefficient for graphs of high magnitudes [40]. Similar to GA and ACO, particle swarm optimization (PSO) has been inspired from the nature and is also a search algorithm that looks for optimum solutions amongst the solutions obtained by each element in the herd. Parameter tuning is necessary for obtaining optimum solutions. PSO performs well in static search spaces and underperforms in dynamic search spaces (that is, the search spaces of the particles are variables) where parameter tuning would need additional heuristics [41]. Simulated annealing (SA) avoids the local optimum very efficiently but lags behind in terms of the accuracy and running time (i.e., computing effort) [42]. ANNs are proven to be very fast, however with lower solution quality [42]; hence, developing a neural network architecture that generates good TSP solutions with less computational complexity is a promising avenue. Hopfield and Tank proposed an evolutionary memory associated neural network for combinatorial optimization, thereby transforming TSP to a memory association problem. But it was shown that the use of Hopfield neural networks (in aforementioned context) for solving NP- complete problems is in fact both memory and time consuming [43]. The Concorde solver is one of the most popular solvers for TSP in recent years [44]. This statement is justified by the tremendous attention that has been devoted to several benchmarks between some new developed concepts and algorithms for solving TSP and the Concorde solver [44]. However, the Concorde solver shows a serious limitation as it is designed for solving symmetric traveling salesman problems (STSP) and thus, does need some conversion/transformation when dealing with the solving of asymmetric traveling salesman problems (ATSP) [45]: indeed the ATSP is the case of undirected graph with different edge costs when travelling in the two contrary directions between two specific adjacent nodes. The use of the Concorde TSP solver for solving ATSP requires a conversion of the ATSP to a symmetric TSP. This conversion is tricky and tedious as it requires additional nodes. The neuro-processor TSP solver concept developed in this paper does efficiently overcome this lastly mentioned problem as it can efficiently solve both STSP and ATSP without any need of transforming, reconfiguring, and/or retraining the neuro-processor solver. This statement is justified by the fact that the neuro-processor solver developed is mathematically modeled by a set of coupled differential equations; the structure of these equations is constant (as it does not vary) and the equations are applicable to graph networks regardless of their size, magnitude and graph topology (say, directed- and/or undirected- graphs, graphs with negative values of weights, graphs with a mixture of negative and positive values of weights, graphs with high/large values of weights, etc.). The coefficients of the coupled differential equations are obtained for a given graph structure/topology. This is one of the main advantages (amongst many others) of using the neuro-processor TSP solver developed in this work. Though the above-mentioned methods (published by the literature) perform sufficiently well in finding approximate solutions for TSP, these methods do not efficiently solve a series of challenges such as real time path/route determination, accurate route detection, finding TSP in dynamically reconfigurable graphs, etc. Since the problem is said to be NP- hard [1], there exist several methods to calculate approximate solutions in polynomial time. By contrast, finding exact accurate solutions encounters convergence as well as running time (i.e., computational effort) issues [46].

C. THE STABILITY CONCEPT AND KEY PERFORMANCE CRITERIA FOR BENCHMARKING

The survey presented in Section I.B. clearly witnesses the tremendous attention devoted to the analysis of TSP with concrete applications in science and engineering. In order to demonstrate the effectiveness of the neuro-processor concept developed for robustly solving TSP in dynamically reconfigurable graph networks, but also for validation purposes, we perform a comprehensive benchmarking, which consists of comparing the results of our novel TSP-solver neuro-processor with those results published in [11]. The key performance criteria considered as benchmarking metrics are:

(a) *Stability of "the previously computed TSP solution/tour"*– it refers to the fact that due to random perturbations of the arc values (values of weights) the total cost of "the newly computed TSP path/tour" is always greater than the total cost of "the previously computed TSP solution/tour" [11].

(b) *Accuracy*– it refers to the sure detection of the true or exact TSP solution.

(c) *Robustness*– it refers to the fact that the performance of the neuro-processor is not affected by the dynamic change of graph-fundamental parameters.

(d) *Flexibility*– it refers to the fact that despite the aforementioned dynamic changes of graph- fundamental parameters, a new (re-) training is not necessary, as this is otherwise (if needed; it is the case for most traditional/related approaches) extremely time-consuming (i.e. very expensive) computationally.

(e) Scalability– it refers to the capability of a solver system (e.g. the neuro-processor developed in this paper) to solve TSP with reasonable/efficient computational effort even for (despite) cases of huge/high graph- sizes and magnitudes. Ideally, the computational effort should remain constant or rise only linearly with raising graph size or magnitude.

Overall, two specific case-study application examples are considered. The first case-study is related to the analysis of the stability of a previously computed optimal TSP tour as reported in [11]: one speaks of "TSP tour stability". This "tour stability" concept refers to its sensitivity to dynamic changes, especially that of arc-weights. Hereby, it is observed which changes in arc-weights are still tolerable without resulting in changing a given optimal TSP tour.

Furthers, the second case-study is related to a comprehensive analysis of the local stability (i.e. the stability of equilibrium/fixed points) of the developed neuro-processor. To the best of our knowledge, this second case-study, which is introduced for the first time concerning TSP solving is a significant novel contribution aiming at demonstrating how the analysis of the stability of equilibrium points can result into the derivation of fundamental relationships/formulas, which can be used to control and ensure/guarantee the fast convergence of the neuro-processor developed to the exact/true TSP solution.

Regarding the ''stability'' analysis conducted in this paper, two different forms of stability are considered.

Consider the first form of stability investigated in this paper, the so-called ''TSP tour stability''. It is the stability of a previously computed tour as reported in [11]. Here, the stability consists of verifying whether the previously computed tour (called optimal TSP path) remains optimal (amongst a set/group of tours) when changes occur in various costs of edges within the graph network. Various stability criteria discussed allow a fast evaluation of whether itineraries remain optimal [11]. Indeed, Ref. [11] develops the integer linear programming (ILP) relaxation method to analyze the stability of ''traveling salesman tours'' when changes occur in various costs of edges within a graph. That thereby conducted analysis considers travel costs between cities for the cases of symmetric TSP (i.e. cases where the arcs' costs are time-homogeneous or time-independent and are also equal for adjacent arcs/edges), asymmetric TSP (i.e. cases where the arc's costs are time-inhomogeneous or time-dependent and are also different for adjacent arcs/edges), and sequence dependent TSP (i.e. cases where the costs of the transition from one city to another depends on both the distance between these cities and on subsequent k cities). Various regions of ''TSP tour'' stability as well as regions of instability are/were thereby computed/determined. Their analysis consists of determining the sensitivity of a given/current optimal TSP-solution to changes in the costs of edges. Indeed, a variation (or monitoring) of the costs of an arbitrary number of edges is performed in order to determine, whether the previously computed TSP tour remains optimal amongst a set/group of tours. The stability is observed whenever the previously computed tour remains optimal, otherwise it is unstable. This statement is used in [11] to derive regions of stability and/or instability in terms of some specific costs of edges, which are used as control parameters (or, to use a system-dynamical term/terminology, bifurcation parameters for stability analysis). Despite the effectiveness of the tour-stability method developed in [11] when dealing with graphs of restricted sizes, the method appears inaccurate when considering graphs of huge/large- sizes [11]. According to this reference, it is challenging (and quite impossible) to enumerate all possible tours when dealing with graphs of huge/large sizes. For such graphs, exact information concerning the ''tour-stability'' cannot be provided using the method in [11].

Consider the second form of stability investigated in this paper, the so-called ''local stability''. It concerns the stability of (the neuro-solver's) equilibrium points (from a system-dynamical perspective), which is further demonstrated as being very critical to guarantee the sure convergence to the exact TSP solution and thereby avoiding faulty TSP detections. Full details of the analytical steps involved in the local stability analysis of TSP related fixed/equilibrium points are provided in Section IV. The focus (in this paper) on the stability of equilibrium points is motivated by the wish of significantly contributing to the enrichment of the related state-of-the-art. Indeed, to the best of our knowledge, the current relevant state-of-the-art does not provide yet a systematic analytical concept to guarantee the sure convergence to the exact TSP solution.

D. CORE CONTRIBUTIONS AND OVERALL PAPER ORGANIZATION

This paper focusses on the mathematical modeling of TSP from a system-dynamical perspective. Analytical expressions /formulas (involving the fundamental parameters of the graph under investigation) are derived (obtained) to guarantee the avoidance of subtours and thus ensure the sole detection of single-cycle TSP-solutions. The paper also comprehensively develops the local stability (also called stability of equilibrium or fixed points) analysis as a systematic analytical concept to predict and control the sure convergence to ''true/exact'' TSP solutions. Furthers, a bifurcation analysis is conducted in order to ensure/guarantee detection of the optimal TSP solution/tour. Analytical relationships (or formulas) are thereby established/derived and used to guarantee both accuracy and robustness of the neuro-processor developed for solving TSP. Overall, the paper develops a systematic analytical concept to tackle/address the five challenging issues/problems (see below in Section II.B) commonly faced (and not satisfactorily solved) by the traditional/related methods, concepts and algorithms for solving TSP. To the best of our knowledge, the current relevant literature does not provide yet a sufficiently comprehensive systematic analytical framework to address the mentioned five challenging issues/problems.

The rest of the paper is organized as follows. Section II explains the modeling principle of traveling salesman problems (TSP) by ordinary differential equations (ODEs). Section III presents full details of the general methodology for modeling TSP through ODEs. Section IV is then concerned with the local stability analysis of the ODE based neuro-processor. Analytical formulas are obtained under which the neuro-processor developed always converges to the exact TSP-solution. Section V focuses on numerical simulations and a validation of the concept(s) presented in Section IV. Two illustrative case-study examples are presented and discussed. The first case-study investigates the stability of a previously computed TSP- tour as reported in [11]. The second case-study is concerned with the numerical study of local stability with the aim of verifying and validating the analytical results obtained in Section IV. Also, the advantage of the local stability analysis for the control and establishment of the convergence properties (of the neuro-processor) to the true/exact TSP solution is clearly demonstrated through various numerical simulations (see bifurcation diagrams in Figs. 8, 9, 10, 11, 15, 19 and 22). The analytical criteria of convergence to the true/exact TSP solution (established in

Section IV) are also validated numerically. To finish, a series of concluding remarks are summarized in Section VI.

II. MOTIVATION, KEY CHALLENGES RELATED TO SOLVING TSP, AND MODELING PRINCIPLE OF TSP

A. MOTIVATION FOR DEVELOPING A TSP SOLVER INSPIRED FROM THE NONLINEAR DYNAMICAL PERSPECTIVE, AND A BRIEF OVERVIEW OF SAMPLE CONTRIBUTIONS EXPLOITING THE BDMM CONCEPT

The literature is very poor in terms of scientific contributions regarding the development of systematic analytical/ comprehensive methods for the efficient solving of TSP in complex (i.e. large) and dynamically reconfigurable graph networks. The nonlinear dynamical perspective used in this paper is based on key concepts such as equilibrium points, stability, convergence, and bifurcation analysis, which are used to develop a systematic analytical framework for a comprehensive addressing of TSP. The analytical framework developed does comprehensively capture all relevant insights of complexity related to the TSP solving. In this context, ordinary differential equations (ODEs) models are found appropriate and convenient modelling instruments. This justifies the modeling procedure that consists of transforming TSP into a constrained optimization problem and thereafter transforming it further into an unconstrained problem through the Lagrange function. Finally, the basic differential multiplier method (BDMM- see Refs. [33], [34], [47]–[49],) is used to obtain the ODE's-based solver model of the TSP. While using the ODE's based solver model of TSP for investigating equilibrium points, stability, convergence and bifurcations, key analytical expressions/conditions are obtained/established for the analysis, handling and/or control of key challenging issues like:

- Dynamic reconfigurability of the TSP solver;
- Stability and/or robustness of the TSP solver;
- Sure convergence to a TSP solution;
- Subtours avoidance and detection of solely single-cycle TSP solutions;

In essence, the TSP modeling procedure encompasses several steps amongst which the BDMM (BDMM: Basis Differential Multiplier Method) constitutes an essential methodological brick in the huge flow-chart (i.e. methodology) describing our TSP modeling procedure. Some illustrative special uses of the BDMM in our previous works at a step of the general optimization procedure can be found in [47]–[49], and in [33], [34] (see also references therein).

Reference [47] develops a new concept based on cellular neural networks (CNN) for the solving of nonlinear ODEs and PDEs. A new templates calculation technique performing nonlinear adaptive optimization (to which we have assigned the acronym of NAOP – nonlinear adaptive optimization process) is used to compute the appropriate cellular neural network templates corresponding to each ODE and/or PDE under investigation. The BDMM is exploited in a step of the procedure for the CNN- template optimization. This procedure encompasses several steps such as: the design of an appropriate CNN- architecture, the formulation of related constraints, the mapping of both dynamical systems to be identical, the use of BDMM, the evaluation of the rate of divergence between the dynamical systems at stake (to be mapped), the CNN-templates calculation, just to name a few.

Reference [48] is focused on the generalization of the NAOP technique developed in [47] for CNN- templates computation. Therefore, a universal concept based on cellular neural networks was developed for the ultrafast and flexible solving of ODEs and PDEs. The main difference between [47] and [48] is the applicability of the concept developed in [48] to all types of complex (i.e., stiff or highly nonlinear) ODEs and PDEs. Similarly, to [47], the BDMM is exploited in a step of the procedure of CNN- templates optimization.

Reference [49] develops a framework based on recurrent neural networks for the efficient solving of both the shortest path problem (SPP) and the shortest path spanning tree problem (SPST). The effectiveness of the framework developed for solving both SPP and SPST was clearly demonstrated through a thoroughly benchmarking of our method with some published methods based on heuristics. Similarly, to the present paper, the BDMM was exploited in a step of the procedure for modeling SPP and SPST through ODEs. Compared to the contribution in this paper, the TSP problem is not addressed in [49]. Also not addressed in [49] is the nonlinear dynamical perspective based on the investigations of issues such as the following ones: equilibrium points, stability, convergence and bifurcation analysis. These issues are addressed in this paper in order to tackle all issues of relevance with regards to the complexity of an efficient and reliable TSP solving.

Reference [33] develops a general optimization concept exploiting the basic differential multiplier method (BDMM). It is thereby demonstrated how BDMM can satisfy the constraints formulated according to a given optimization problem. In essence, the constrained optimization problem is transformed into an unconstrained optimization problem characterized by the Lagrange function expressing the total energy of the system. The Lagrange function is expressed in terms of both "decision neurons" and "multiplier neurons". While using BDMM, a set of differential equations is obtained to model an optimization problem at stake. As proof of concept (in order to validate the results), two concrete application examples are considered: the case of enforcing permutation codewords in the analog decoding problem, and the case of enforcing valid tours in TSP. Compared to the contribution in this paper, no systematic analytical method is provided in [33], specifically to address the TSP problem. Furthers, the nonlinear dynamical perspective involving the investigation of issues like equilibrium points (fixed points), stability, convergence and bifurcation analysis is not addressed. Let's note once more that the nonlinear dynamical perspective is very important to tackle/handle all relevant aspects with regards to the complexity and the reliability related to solving TSP.

Reference [34] develops an algorithm based on neural networks to solve the multiple traveling salesmen problem (MTSP). The algorithm developed is expressed in the form of differential equations. These equations are obtained by combining the basic differential multiplier method (BDMM) [33] with an expanded version of Hopfield-Tank's neuromorphic city-position map [35]. The MTSP is modeled into the form of an energy function describing the total length (to be minimized) of a tour. This tour is composed of multiple non-trivial subtours with a common node/vertex representing the base city. Here, nontrivial subtours are subtours with at least one city other than the base city. The great advantage of the algorithm developed is clearly justified in [34]. Indeed, the algorithm expressed in the form of differential equations can be implemented on hardware and thus solutions to complicated decision-making problems are (or can be) obtained by solving the differential equations in (real-) time. This comment witnesses the potential of achieving very good computing performances when solving MTSP and TSP using algorithms based on differential equations. It is also clearly stated in [34] that ODEs-based models are easily implementable on analog circuits characterized by an ultra-fast computing performance. By contrast thereto, in [34], the main drawbacks of the Hopfield and Tank's method are discussed, amongst which the stability of both MTSP and TSP solutions as well as the optimality of their solutions are clearly underscored, which are important issues not addressed so far in [34]. The solution proposed in [34] to tackle/overcome the aforementioned drawbacks was to combine Hopfield and Tank's methods with the BDMM. By combining these last-mentioned methods, it leads to the algorithm developed and presented in [34] for the solving of both MTSP and TSP. A further interesting concluding remark was formulated in [34] (see the last paragraph of section 5). Indeed, it is thereby clearly stated that the stability of the algorithm developed in [34] for solving MTSP and TSP as well as the optimality of the respective solutions are important (explicitly not, resp. not yet solved/addressed) research questions (or research avenues) to be further investigated in future research, as outlooks. This comment can be used to witness/justify the key contribution of our work addressing amongst many other questions, these last-named two research questions/issues formulated in [34] as outlooks. When comparing the method/approach in [34] with our method in this paper, the key difference and/or key innovation is summarized by the five bullets/points below, which are comprehensively and explicitly considered in this work, while, however, they were not addressed in [34]:

- The mathematical modeling of subtours and the respective subtours avoidance;
- The new mathematical modeling of the so-called binarization constraints (i.e.: they do express whether a given edge does belong to the final TSP solution or not);
- The analytical investigation of stability and convergence properties of the single-cycle TSP solver model developed;
- A comprehensive bifurcation analysis and the derivation of conditions/ranges of parameters to ensure the "always"-optimality of detected single-cycle TSP solutions;
- The possibility of detecting only single-cycle TSP solutions. This corresponds to one salesman and one base city (also called "TSP solution with a single depot").

B. COMPLEXITY, STABILITY, AND ROBUSTNESS: KEY CHALLENGES RELATED TO SOLVING TSP

According to the overview provided above, it clearly appears that nowadays several challenging issues/problems still exist (see below), which are not yet satisfactorily solved/addressed by existing methods for solving TSP:

1) Lack of flexibility.

With respect to dynamic changes in the fundamental parameters of a graph network (e.g. arc's weights, size, magnitude, topology, etc.) most existing methods (concepts and algorithms) do need either a new (re-) training or a new start of calculations, which consequently result in additional significant computational costs. By contrast, the mathematical model of the neuro-processor solver developed here is expressed by a model made of coupled ordinary differential equations (ODEs). The main advantage is that the coupled ODE-model remains valid and thus applicable to all changes in the graph fundamental parameters. The flexibility of the neuro-processor solver developed is therefore justified as the dynamic changes in the graph- fundamental parameters (i.e., magnitude, size, costs of edges, and graph topology) affect solely the coefficients of the coupled ODE-model (and these belong to the model/system inputs, which can freely dynamically change (over time)) and do not affect the structure (or the form) of the ODE-model describing the neuro-processor. The advantage hereby is that the output of the neuro-processor is obtained through a solution of the (dynamically reconfigurable – through inputs) coupled ODE-model. Therefore, no re-training is needed as the numerical solving of ODEs is realized as a direct iterative process even in case of ODEs models with time-varying coefficients.

2) Weak robustness to the dynamic change of graph-fundamental parameters (e.g. values of weights, size, magnitude, and topology).

In the concept of this paper, the robustness (this is due to the fact that the performance of the neuro-processor is not affected by the dynamic change of graph parameters "size, magnitude, and topology, etc.") of the neuro-processor TSP solver developed is ensured by a system-dynamical perspective.

In fact, an appropriate analytical formulation of the key modeling constraints is comprehensively done (e.g. connectivity of nodes (see Eq. (10)), binarization (see Eq. (11)), subtours avoidance (see Eq. (13)) and also, the possibility of monitoring the step sizes of both decision and multiplier neurons (see α, β_i, β_3^*, and $\beta_{5,l}$ in Eq. (15)). Furthers, a straight-forward comprehensive demonstration involving some fundamental instruments/concepts from the nonlinear system-dynamical science (e.g. stability analysis, bifurcation analysis, etc.) are used to demonstrate an ''always''-sure convergence to the exact/true (single-cycle) solution despite the dynamical changes of graph parameters. This demonstration and related validation is performed through various numerical simulations in Section V.

3) Poor capacity of avoiding subtours and thus the poor capacity of ensuring the detection of single-cycle TSP- tours.

In case of complex graphs, several classical/traditional algorithms do generally detect TSP-solutions in form of subtours (called multiple TSP (MTSP)) rather than detecting TSP- solutions/tours with only a single-cycle.

The modeling of subtours is carried out (in this paper) and analytical expressions (involving the fundamental parameters of the graph under investigation) are used/involved in the overall concepts as constraints in order to ensure the avoidance of multiple TSP (i.e., MTSP) solutions and therefore ensure the ''always''-convergence to single-cycle TSP solutions.

4) Failure to converge during the TSP solving.

It is known (from the literature) that several classical/traditional methods and algorithms often do not converge in cases of the simultaneous presence of several optimal TSP-tours called global minima (with different trajectories of equal total cost) co-exist in a graph. The investigation of the respective stability (stability of the neuro-processor from a system-dynamical perspective) is carried out in this paper to establish key analytical conditions/formulas to ensure/guarantee the ''always''-convergence to an optimal TSP solution even in cases where several optimal TSP-solutions (with different trajectories) coexist in a graph. The analytical/theoretical results obtained (in Section IV) are confirmed (in Section V) through various numerical simulations.

5) Failure to detect the true/exact TSP solution/tour.

The true/exact TSP solution also called ''optimal TSP solution'' is the one corresponding to the global minimum. This challenging problem (i.e. failure to detect the true global optimum) generally occurs whenever several local minima co-exist in the vicinity (neighborhood) of the global minimum.

A bifurcation analysis conducted in this paper is used to demonstrate that appropriate windows/ranges for values/settings of the neuro-processor's parameters can be selected and fixed to ensure an ''always''-optimality of the TSP solution despite/regardless of the specific settings of various input graphs and their related respective parameters' settings (i.e. arc weights, topology, etc.). This demonstration is explained, validated, and experimentally proven (in Section V) through various numerical simulations.

C. MODELING PRINCIPLE OF TSP THROUGH ODES

This section focuses on the full description of the developed concept for traveling salesman optimal path/tour detection problem in dynamically reconfigurable graphs. The synoptic representation of the concept is depicted in Figure 1.

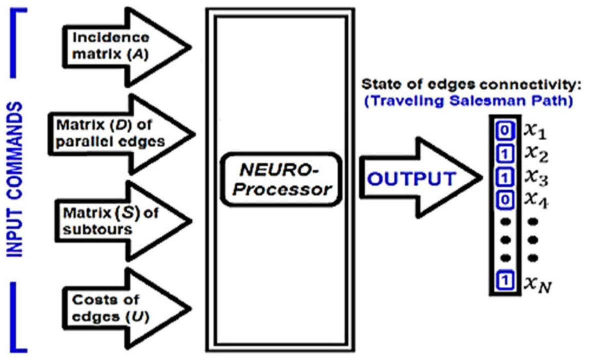


FIGURE 1. Synoptic representation of the neuro-processor based TSP solver concept. The four inputs are fundamental parameters of the graph under investigation. These inputs are used as external commands. The output $x_i = 1$ expresses the belonging of an edge to the TSP solution, and $x_i = 0$ otherwise.

The neuro-processor solver is made-up of four inputs A, D, S, and U (see Figure 1). The first three inputs are matrices and the fourth input (U) is a vector. The details of these inputs are provided in Sections III and IV. The output x_i of the neuro-processor is binary; $x_i = 1$ expresses the belonging of an edge with state x_i to the TSP solution and $x_i = 0$ otherwise. The core of the neuro-processor is modeled by a set of coupled ODEs. The coefficients of the coupled ODEs-model are the inputs (A, D, S, and U), and the step sizes of both decision and multiplier neurons are denoted by α and β_i respectively (these are internal parameters of the neuro-processor; more details are provided in Sections III and IV).

The main advantage of the neuro-processor based TSP solver concept developed is that the inputs are made-up of the fundamental parameters of the graph under investigation. These inputs are therefore used as external commands, which can even dynamically change without affecting the neuro-processor. Besides, the coefficients of the coupled ODEs-model are expressed in matrix form and this expression witnesses the scalability of the coupled ODEs-model. Also, the coupled ODE-model is a suitable framework for both stability and bifurcation analyses (from a system-dynamical perspective).

In some of our recent papers, the neuro-processor based concept has been successfully applied for solving nonlinear ordinary and partial differential equations (ODEs and PDEs) [47], [48] and also for solving shortest path problems (SPP) and shortest path tree problems (SPTP) in dynamically reconfigurable graph/networks (see [49]). Worth mentioning that the neuro-processor based TSP solver is of general applicability as it has already been successfully applied to various graphs, regardless of their sizes, magnitudes, structures/topologies, the weight-values (e.g. low costs values, high costs values, negative costs, positive costs, mixture of negative and positive costs, etc.) (See our Ref. [49]). By contrast, it is well-known that the robustness of heuristic methods is drastically affected for high weights values [49]. This statement is justified by the benchmarking performed in this paper (see Section V). This benchmarking could be used to underscore the neuro-processor based TSP solver concept developed as a good candidate to overcome the inherent serious drawbacks (e.g. instability and weak robustness) of most heuristic (traditional) methods for TSP solving.

III. GENERAL METHODOLOGY FOR MODELING TSP: LAGRANGE FUNCTION AND COUPLED ODES MODEL

A. TRANSFORMING TSP INTO AN OPTIMIZATION PROBLEM

1) DERIVATION OF THE LAGRANGIAN AS "TOTAL ENERGY" OF THE SYSTEM (SEE EQ. (7) AND ALSO [47]–[49])

Considering the TSP as a nonlinear optimization problem, we define the objective function by Eq. (1), where the function $f(\vec{x})$ represents the total cost of the graph.

$$Min\left[f(\vec{x})\right] \tag{1}$$

The objective function (1) is subject to constraints (2) - (6).

$$\begin{cases} g_{1j}(\vec{x}) = 0 & (2) \\ g_{2j}(\vec{x}) = 0 & (3) \end{cases}$$

$$\begin{cases} g_{3i}(\vec{x}) = 0 & (4a) \\ g^*_{3i}(\vec{x}) = 0 & (4b) \end{cases}$$

$$\begin{cases} g_{4i}(\vec{x}) = 0 & (5) \\ g_{5k,l}(\vec{x}) = 0 & (6) \end{cases}$$

Note that (4) are split up into (4a) and (4b) because they are mathematically equivalent. However, in the context of the neuro-processor TSP solver developed, (4a) and (4b) are used simultaneously to ensure a robust binarization of components of $\vec{x}$ (see details below) characterized by two stable equilibrium points $P_1(x = 0)$ and P_2 $(x = 1)$ as shown in Figure 2.

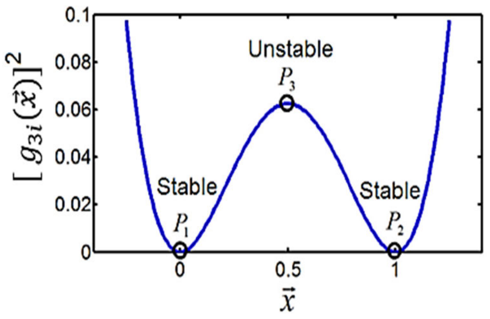


FIGURE 2. Illustration of the binarization constraints. The constraints $g^*_{3i}(\vec{x}) = [g_{3i}(\vec{x})]^2 = 0$ are fulfilled at points $P_1(x = 0)$ and P_2 $(x = 1)$. The constraint $g_{3i}(\vec{x}) = \vec{x}(\vec{x} - 1)$ is squared to obtain the two stable equilibrium points P_j.

In Equation (1) to (6), the vectors of multiplier variables are expressed as follows:

$$\vec{\lambda} = [\lambda_1, \lambda_2, \ldots, \lambda_M]^T; \quad \vec{\varphi} = [\varphi_1, \varphi_2, \ldots, \varphi_M]^T;$$
$$\vec{\gamma} = [\gamma_1, \gamma_2, \ldots, \gamma_N]^T; \quad \vec{\gamma}^* = \left[\gamma^*_1, \gamma^*_2, \ldots, \gamma^*_N\right]^T;$$
$$\vec{\omega} = \left[\omega_1, \omega_2, \ldots, \omega_{N/2}\right]^T; \quad \vec{\mu}_l = \left[\mu_{1,l}, \mu_{2,l}, \ldots, \mu_{\aleph,l}\right]^T;$$

their components are denoted $\lambda_j, \varphi_j, \gamma_i, \gamma^*_j, \omega_i$, and $\mu_{k,l}$. The decision variable $\vec{x} = [x_1, x_2, \ldots, x_N]^T$ is the state vector of all edges x_i of the graph under consideration ($x_i = 1$ for edges belonging to the TSP solution and $x_i = 0$ otherwise). Thus, the dimension of $\vec{x}$ is equal to N (N is the graph size). The optimization constraints $g_{1j}(\vec{x}) = 0$ and $g_{2j}(\vec{x}) = 0$ (see (10)) are formulated to ensure the belonging of each node of the graph to the TSP solution. Constraints $g_{3i}(\vec{x}) = 0$ and $g^*_{3i}(\vec{x}) = 0$ (see (11)) ensure the binarization of all components of vector $\vec{x}$. These constraints are identical mathematically. However as depicted in Figure 2, the expression $g^*_{3i}(\vec{x}) = \left[g_{3i}(\vec{x})\right]^2$ is characterized by a double-well potential with two stable equilibrium points $P_1(x = 0)$ and P_2 $(x = 1)$. Thus, using constraints $g_{3i}(\vec{x})$ and $g^*_{3i}(\vec{x})$ simultaneously leads to consolidate the binarization and this ensures a robust binarization of all components of $\vec{x}$. The advantage of using (4a) and (4b) simultaneously is also reported in our previous related works/references [47], [48]. Constraints $g_{4i}(\vec{x}) = 0$ (see (12)) are formulated to avoid adjacent edges in the TSP solution, specifically in the general case of undirected graph networks. Finally, constraints $g_{5k,l}(\vec{x}) = 0$ (see (13)) are introduced to avoid subtours in the TSP solution and therefore ensure solely convergence to a single-cycle (one-cycle) TSP solution.

According to the above definitions of all optimization constraints, the indexes i, j, k, and l are defined as follows: $i = 1, 2, \ldots, N$, $j = 1, 2, \ldots, M$, $k = 1, 2, \ldots, \aleph$, and $l = 1, 2, \ldots, n$. The integers $M, N, \aleph$ and n represent respectively the magnitude, size, the total number of subtours in a given group of subtours S_k, and the total number of different groups of subtours identified in a graph G. Therefore, in the last term of (7), the index l is used for the numbering of constraints in each group of subtours (say, $g_{5k,1}(\vec{x})$ denotes the constraints in group 1 and $g_{5k,2}(\vec{x})$ denotes the constraints in group 2, etc.). Similarly, the index k denotes the subtours found in a given group of subtours (say, $k = 1, \ldots, 20$ in a class/group of TSP solutions with a total of 20 subtours; $k = 1, \ldots, 500$ in a class/group of TSP solutions with a total of 500 subtours, etc.). Details regarding the identification of subtours are provided below (see section related to the modeling of subtours); more details can also be found in [50], [51].

The relaxation method is used to transform the constrained optimization problem (modeled by (1) to (6)) into the unconstrained optimization problem (see (7)).

$$\begin{aligned}\tilde{L} &= f(\vec{x}) + \sum\nolimits_{j=1}^{M} \lambda_j \cdot \left[g_{1j}(\vec{x})\right] \\ &\quad + \sum\nolimits_{j=1}^{M} \varphi_j \cdot \left[g_{2j}(\vec{x})\right] + \sum\nolimits_{i=1}^{N} \gamma_i \cdot \left[g_{3i}(\vec{x})\right] \\ &\quad + \sum\nolimits_{i=1}^{N} \gamma^*_i \cdot \left[g^*_{3i}(\vec{x})\right] + \sum\nolimits_{i=1}^{\frac{N}{2}} \omega_i \cdot \left[g_{4i}(\vec{x})\right] \\ &\quad + \sum\nolimits_{l=1}^{n} \sum\nolimits_{k=1}^{\aleph} \mu_{k,l} \cdot \left[g_{5k,l}(\vec{x})\right]\end{aligned} \tag{7}$$

2) APPLICATION OF THE BASIC DIFFERENTIAL MULTIPLIER METHOD (BDMM) [33], [34], [47]–[49]

The concept of neuron-dynamics is exploited as an optimization strategy that maps the optimization problem unto the energy of a neural network in order to find the optimal solution. The underlined energy is represented by the Lagrange function as the total energy of the system. Thus, minimizing the Lagrange function leads to a stable state. This stability is explained by the sure convergence of the basic differential multiplier method (BDMM). An in-depth demonstration of this convergence can be found in our previous works, see recently published papers [47]–[49]. Thus, applying the BDMM to the Lagrange function in (7) leads to (8).

$$\begin{cases} \frac{dx_i}{dt} = -\alpha \frac{\partial L}{\partial x_i} \\ \frac{d\lambda_j}{dt} = \beta_1 \frac{\partial L}{\partial \lambda_j}; \ \frac{d\varphi_j}{dt} = \beta_2 \frac{\partial L}{\partial \varphi_j}; \ \frac{d\gamma_i}{dt} = \beta_3 \frac{\partial L}{\partial \gamma_i} \\ \frac{d\gamma_i^*}{dt} = \beta_3^* \frac{\partial L}{\partial \gamma_i^*}; \ \frac{d\omega_i}{dt} = \beta_4 \frac{\partial L}{\partial \omega_i}; \ \frac{d\mu_{k,l}}{dt} = \beta_{5,l} \frac{\partial L}{\partial \mu_{k,l}} \end{cases} \tag{8}$$

In Equation (8), x_i are components of the vector of decision variables $\vec{x}$ $(i = 1, 2, \ldots N)$, λ_j are components of the vector of multiplier variables $\vec{\lambda}$ $(j = 1, 2, \ldots M)$, φ_j are components of the vector of multiplier variables $\vec{\varphi}$ $(j = 1, 2, \ldots M)$, γ_i and γ_i^* are components of the vectors of multiplier variables $\vec{\gamma}$ $(i = 1, 2, \ldots N)$ and $\vec{\gamma}^*$ $(i = 1, 2, \ldots N)$, ω_i are components of the vectors of multiplier variables $\vec{\omega}$ $(i = 1, 2, \ldots N/2)$ and $\mu_{k,l}$ are components of the vector of multiplier variables $\vec{\mu}_l$ $(k = 1, 2, \ldots \aleph$ and $l = 1, 2, \ldots n)$. As already mentioned above and due to the importance of these two parameters (k and l) let's recall here that the parameter k is the index of each subtour and $\aleph$ is the total number of subtours in a specific class/group of subtours. l is the index assigned to each class/group of subtours and n is the total number of classes/groups of subtours in the graph. It is worth mentioning that according to (8) the variables x_i, λ_j, φ_j, γ_i, γ_i^*, ω_i, and $\mu_{k,l}$ depend on "t" ("t" is the independent variable of the set of (8)). Further, the set of (8) reveals the coupling between decision variables x_i and the multiplier variables λ_j, φ_j, γ_i, γ_i^*, ω_i, and $\mu_{k,l}$. These dependent variables converge to equilibrium/fixed points (as "t" increases): this convergence corresponds to the end of the optimization process. The quantities α and β_i represent respectively the step sizes for updating decision- and multiplier- variables. These quantities are positive real numbers ($\alpha > 0$; $\beta_i > 0$). The next section (Section III. B.) shows how the mathematical model of the neuro-processor based TSP solver concept can be obtained/derived using (8).

B. MATHEMATICAL MODEL OF THE NEURO- PROCESSOR BASED TSP SOLVER SIMULATOR

1) FORMULATION OF CONSTRAINTS WITH REGARDS TO THE TSP

Let's consider a graph denoted by $G = (V, E)$, where V and E represent the sets of nodes and edges respectively. The objective function is formulated as $Min\left[f(\vec{x})\right]$ where $f(\vec{x})$ is the total cost of G. In (9) the costs of edges are denoted c_i and the decision variables x_i give the states of edges: $x_i = 1$ for edges belonging to the TSP solution and $x_i = 0$ otherwise.

$$f(\vec{x}) = \sum_{i=1}^{N} c_i x_i \begin{cases} x_i \in F = \{0, 1\} \\ \forall i | c_i \in \Re \end{cases} \tag{9}$$

The expression (9) does assume a linear path cost model. Nevertheless, in various practical cases the path cost may be nonlinear. This latter issue is to be addressed in future works.

The constraint to ensure the connectivity of all nodes (belonging to the optimal TSP route/path) is modeled mathematically by (10). The system (10) is applied to each node j $(j = 1 \ldots M)$ of the graph under investigation.

$$\text{At node } j: \begin{cases} g_1(\vec{x}) = \sum_{\substack{i=1 \\ i \neq j}}^{N} x_i * h_{ij} = 0 \\ g_2(\vec{x}) = \sum_{\substack{i=1 \\ i \neq j}}^{N} x_i - 2 = 0 \end{cases} \tag{10}$$

The parameter h_{ij} in (10) can take only two values ($h_{ij} = -1$ or $h_{ij} = +1$). At a node j, 2 groups of edges are identified and a specific value of h_{ij} is assigned to edges belonging to a particular group. Specifically, $h_{ij} = -1$ for all incoming edges (i.e., edges entering the node j) and $h_{ij} = +1$ for all outgoing edges (i.e., edges leaving the node j).

The constraints formulated in (11) ensure the binarization of state variables (decision variables) x_i of all edges of graph G.

$$g_3(\vec{x}) = x_i(x_i - 1) = 0 \tag{11a}$$

$$g_3^*(\vec{x}) = [x_i(x_i - 1)]^2 = 0 \tag{11b}$$

The constraints (11) are reported in our previous works published in [47] and [48].

The constraints (12) are used in case of bi-directed graph. Here, vertices or nodes are pairwise connected with two edges in opposite directions (called parallel or adjacent edges). Thus, constraints (12) are formulated to avoid the belonging of parallel edges to the TSP solution. Considering a given pair of vertices/nodes connected by two parallel edges x_{2i} (in one direction) and x_{2i-1} (in reverse direction), constraints (12a) are formulated accordingly (where, x_{2i} and x_{2i-1} are binary):

$$(x_{2i-1} + x_{2i})(x_{2i-1} + x_{2i} - 1) = 0 \tag{12a}$$

The expression (12a) avoids the belonging of a pair of parallel edges to the TSP solution/tour. Therefore, only one edge of a pair of parallel edges can belong to the TSP solution/tour). Applying (12a) to all parallel (or adjacent) edges leads to the following expression into matrix-form (D is defined in (15h), at the bottom of page 12):

$$g_4(\vec{x}) = D\vec{x}^{\circ}(D\vec{x} - 1) = 0 \tag{12b}$$

The matrix D is easily obtained for a given graph as follows: D_1 is the pair of parallel edges (x_1, x_2). D_2 is the pair of parallel edges (x_3, x_4), and $D_{N/2}$ is the pair of parallel edges (x_{N-1}, x_N). Thus the matrix D in (15h) is of dimension $\left(\frac{N}{2}\right) \times N$, where N is the size of the graph under investigation. The vector $\vec{x} = [x_1, x_2, \ldots, x_N]^T$ is the state vector of edges in G and the operator "$\circ$" performs the elementwise product.

The constraint (13a) is formulated to avoid the belonging of a particular class/group of subtours to the TSP solution. Specifically, the constraint (13a) avoids the belonging of subtours with exactly $\mathbb{N}$ edges to the TSP solution. x_j are the edges forming a given subtour in S_k. When (13a) is fulfilled the $\mathbb{N}$ edges of a subtour in S_k cannot form a closed loop (cycle).

$$g_{5k,l}(\vec{x}) = \prod_{i=1}^{\mathbb{N}} \left[\sum_{\substack{j=1 \\ x_j \epsilon S_k}}^{\mathbb{N}} (x_j - i + 1) \right] = 0 \qquad (13a)$$

Considering a specific class/group of subtours S_k with exactly $\mathbb{N}$ edges each, if we denote by $\aleph$ the total number of subtours identified in the class/group of subtours, thus the constraint (13a) can be applied to all subtours leading to the general expression in (13b).

$$\sum_{k=1}^{\aleph} \mu_k \cdot \left[\prod_{i=0}^{(\mathbb{N}-1)} \left(\sum_{\substack{j=1 \\ x_j \epsilon S_k}}^{\mathbb{N}} x_j - i \right) \right] = 0 \qquad (13b)$$

For a given class/group of subtours, μ_k are coefficients assigned to each subtour in the set of subtours S_k containing a total number of $\aleph$ subtours with exactly $\mathbb{N}$ edges each.

We now consider different classes/groups of subtours. For example: Group 1 (subtours with 3 edges ($\mathbb{N} = 3$)), Group 2 (subtours with 4 edges ($\mathbb{N} = 4$), Group n (subtours with N edges)), etc. For the sake of generalization we introduce a new parameter $l = 1, \ldots, n$ to denote different classes/groups of subtours identified in a graph G under investigation. Thus applying (13b) to all classes/groups leads to expression (13c).

$$\sum_{l=1}^{n} \sum_{k=1}^{\aleph} \mu_{k,l} \cdot \left[\prod_{i=0}^{(\mathbb{N}-1)} \left(\sum_{\substack{j=1 \\ x_j \epsilon S_k}}^{\mathbb{N}} x_j - i \right) \right] = 0 \qquad (13c)$$

Worth mentioning that the constraints (13) assume a prior knowledge of subtours in a graph G under investigation. This can be done using existing/classical algorithms. Indeed, several algorithms do exist for the determination of subtours in a graph G (see Refs. [50], [51] and references therein). Further worth mentioning that the subtours do not consider the real values of arc weights. In fact the elements of the matrix S of subtours can take only two values ("0" and "1"). As it appears from the matrix S of subtour in (15h), the edges x_i belonging to a subtour S_k are denoted in the matrix S by the value "1" and the edges which do not belong to S_k are denoted by "0".

In (15h) we have represented only the matrix S of the class/group of subtours S_k containing exactly 3-edges each. Similarly the matrices S of the other classes/groups of subtours S_k (say, a class/group of subtours containing exactly 4-edges, 5-edges, and up to $\mathbb{N}$-edges) can be easily deduced according to (15h). More precisely, considering the number of elements with values "1" in a row of matrices S, this number corresponds to the number of edges forming each subtour in a class/group of subtours S_k.

To illustrate the applicability of the general expression (13a) we consider the following concrete case of subtours formed by exactly 3-edges (e.g., (x_1, x_2, x_3), (x_4, x_5, x_6), (x_7, x_8, x_9), …, *etc.*). According to (13a) the following set of algebraic equations is obtained:

$$\begin{cases} (x_1 + x_2 + x_3)(x_1 + x_2 + x_3 - 1)(x_1 + x_2 + x_3 - 2) = 0 \\ (x_4 + x_5 + x_6)(x_4 + x_5 + x_6 - 1)(x_4 + x_5 + x_6 - 2) = 0 \\ (x_7 + x_8 + x_9)(x_7 + x_8 + x_9 - 1)(x_7 + x_8 + x_9 - 2) = 0 \\ \qquad \ldots etc \ldots \end{cases}$$

Therefore considering the class/group of subtours containing exactly 3-edges (within a graph G) and applying (13a) leads to the following general expression into matrix-form:

$$g_{5k,l}(\vec{x}) = S\vec{x}^{\circ}(S\vec{x} - 1)^{\circ}(S\vec{x} - 2) = 0 \qquad (13d)$$

S is a matrix of size $\aleph \times N$, where $\aleph$ is the total number of subtours in a given class/group and N is the size of the graph.

For example, $\mathbb{N} = 3$ and $\aleph = 20$ if the graph G contains twenty subtours S_k, each with exactly 3-edges, $\mathbb{N} = 6$ and $\aleph = 50$ if G contains fifty subtours S_k, each with exactly 6-edges, etc. This example clearly shows that the parameters $\mathbb{N}$ and $\aleph$ are to be determined for each class/group of subtours S_k. As already mentioned above, this can be done/achieved through the use of existing classical methods and concepts [50], [51].

Let's note that matrix S is determined/defined for each class/group of subtours. The columns of matrix S correspond to the edges of the graph under investigation and the lines/rows of matrix S correspond to the subtours S_k, which are identified in a given class/group of subtours. This remark is essential as it clearly shows that different matrices S are obtained, each of which corresponds to a specific class/group of subtours S_k.

As generalization of the constraint (13d) the case of $\aleph$ subtours, belonging to a given group of subtours with exactly $\mathbb{N}$ edges is modeled by (13e). As already explained the size of S depends on $\aleph$ (dimension of rows) and N (dimension of columns). Further the elements ("0" / "1") in S depend on $\mathbb{N}$.

$$g_{5k,l}(\vec{x}) = S\vec{x}^{\circ}(S\vec{x} - 1)^{\circ} \ldots ^{\circ}(S\vec{x} - \mathbb{N} + 1) = 0 \qquad (13e)$$

It is worth mentioning here that the constraint $g_{5k,l}(\vec{x})$ in Eq. (13e) is defined/determined for each class/group of subtours S_k. Thus (13e) is a general expression/formulation applicable to all classes/groups of subtours S_k. The expression (13e) is consequently written for each class/group of subtours S_k.

2) EXPRESSION OF THE LAGRANGE FUNCTION AS TOTAL ENERGY OF THE SYSTEM AND DERIVATION OF THE MATHEMATICAL MODEL OF THE NEURO-PROCESSOR FOR TSP SOLVING

We now substitute constraints (10) to (13) into (7) to obtain the Lagrange function in (14) as total energy of the system.

$$\begin{aligned}
&\tilde{L}\left(\vec{x},\vec{\lambda},\vec{\varphi},\vec{\gamma},\vec{\omega},\vec{\mu}_l\right)\\
&=\sum_{i=1}^{N} c_i x_i+\sum_{j=1}^{M}\lambda_j\cdot\left(\sum_{\substack{i=1\\ i\neq j}}^{N} x_i * a_{ij}\right)\\
&\quad+\sum\nolimits_{j=1}^{M}\varphi_j\cdot\left(\sum\nolimits_{\substack{i=1\\ i\neq j}}^{N} x_i-2\right)\\
&\quad+\sum\nolimits_{i=1}^{N}\gamma_i\cdot\left[x_i\left(x_i-1\right)\right]+\sum\nolimits_{i=1}^{N}\gamma_i^{*}\cdot\left[x_i\left(x_i-1\right)\right]^2\\
&\quad+\sum\nolimits_{i=1}^{\frac{N}{2}}\omega_i\cdot\left[\left(x_{2i-1}+x_{2i}\right)\left(x_{2i-1}+x_{2i}-1\right)\right]\\
&\quad+\sum\nolimits_{l=1}^{n}\sum\nolimits_{k=1}^{\aleph}\mu_{k,l}\cdot\left[\prod\nolimits_{i=0}^{(\mathbb{N}-1)}\left(\sum\nolimits_{\substack{j=1\\ x_j\epsilon S_k}}^{\mathbb{N}} x_j-i\right)\right]
\end{aligned}\tag{14}$$

In (14), the quantities c_i, x_i, λ_j, φ_j, γ_i, γ_i^*, ω_i, and $\mu_{k,l}$ are components of vectors $\vec{U}$, $\vec{x}$, $\vec{\lambda}$, $\vec{\varphi}$, $\vec{\gamma}$, $\vec{\gamma}^*$, $\vec{\omega}$ and $\vec{\mu}_l$ respectively. These vectors, respectively with dimensions N, N, M, M, N, N, $N/2$ and $\aleph$ are defined as follows:

$$\begin{aligned}
&\vec{U}=\left[c_1,c_2,\ldots,c_{N-1},c_N\right]^T;\quad \vec{x}=\left[x_1,\ldots,x_{N-1},x_N\right]^T;\\
&\vec{\lambda}=\left[\lambda_1,\lambda_2,\ldots,\lambda_{M-1},\lambda_M\right]^T;\quad \vec{\varphi}=\left[\varphi_1,\ldots,\varphi_{M-1},\varphi_M\right]^T;\\
&\vec{\gamma}=\left[\gamma_1,\gamma_2,\ldots,\gamma_{N-1},\gamma_N\right]^T;\quad \vec{\gamma}^*=\left[\gamma_1^*,\ldots,\gamma_{N-1}^*,\gamma_N^*\right]^T;\\
&\vec{\omega}=\left[\omega_1,\ldots,\omega_{N/2-1},\omega_{N/2}\right]^T;\\
&\vec{\mu}_l=\left[\mu_{1,l},\ldots,\mu_{\aleph-1,l},\mu_{\aleph,l}\right]^T.
\end{aligned}$$

The Lagrange function $\tilde{L}\left(\vec{x},\vec{\lambda},\vec{\varphi},\vec{\gamma},\vec{\gamma}^*,\vec{\omega},\vec{\mu}_l\right)$ in (14) is now substituted into the expression of BDMM in (8). Performing the partial derivatives in all directions (or in all components) of the dependent variables $\vec{x}$, $\vec{\lambda}$, $\vec{\varphi}$, $\vec{\gamma}$, $\vec{\gamma}^*$, $\vec{\omega}$ and $\vec{\mu}_l$ leads to the derivation of (15). More details concerning the calculation of all partial derivatives can be found in [49]. Equation (15) is the mathematical model of the neuro-processor based TSP solver concept developed in this paper.

In Equation (15g) the parameter $\beta_{5,l}$ corresponds to a given class/group of subtours (e.g., $\beta_{5,1}$ for subtours in group-1; $\beta_{5,2}$ for subtours in group-2; $\beta_{5,n}$ for subtours in group-n). Therefore (15g) is written for each class/group of subtours.

$$\begin{aligned}
\frac{d\vec{x}}{dt}&=-\alpha\Big\{A^T\vec{\lambda}+A^{+T}\vec{\varphi}+\left(2\vec{x}-1\right)\circ\vec{\gamma}\\
&\quad+\left(4\vec{x}^3-6\vec{x}^2+2\vec{x}\right)\circ\vec{\gamma}^*+\left(2D^TD\vec{x}-1\right)\circ D^T\vec{\omega}+\vec{U}\\
&\quad+\sum_{l=1}^{n}\Big[\left(S^TS\vec{x}\right)^{\circ}\left(S^TS\vec{x}-1\right)^{\circ}\\
&\quad\ldots^{\circ}\left(S^TS\vec{x}-\mathbb{N}+1\right)\Big]\times\Big[\left(S^TS\vec{x}\right)^{-1}+\left(S^TS\vec{x}-1\right)^{-1}\\
&\quad+\ldots+\left(S^TS\vec{x}-\mathbb{N}+1\right)^{-1}\Big]\circ S^T\vec{\mu}_l\Big\}
\end{aligned}\tag{15a}$$

$$\frac{d\vec{\lambda}}{dt}=\beta_1\left[A\vec{x}\right]\tag{15b}$$

$$\frac{d\vec{\varphi}}{dt}=\beta_2\left[A^{+}\vec{x}-2\right]\tag{15c}$$

$$\frac{d\vec{\gamma}}{dt}=\beta_3\vec{x}\circ\left(\vec{x}-1\right)\tag{15d}$$

$$\frac{d\vec{\gamma}^*}{dt}=\beta_3^*\left[\vec{x}^4-2\vec{x}^3+\vec{x}^2\right]\tag{15e}$$

$$\frac{d\vec{\omega}}{dt}=\beta_4\left[D\vec{x}\circ\left(D\vec{x}-1\right)\right]\tag{15f}$$

$$\frac{d\vec{\mu}_l}{dt}=\beta_{5,l}\left[S\vec{x}^{\circ}\left(S\vec{x}-1\right)^{\circ}\ldots^{\circ}\left(S\vec{x}-\mathbb{N}+1\right)\right]\tag{15g}$$

The fundamental parameters of (15) are the coefficients defined as follows: the matrix A and its transpose A^T; the absolute value of A denoted by A^+ and its transpose A^{+T}; the matrix D and its transpose D^T; the matrices S (each of which corresponds to a specific class/group of subtours S_k); the transpose of matrices S denoted by S^T. As already explained above the matrix S is defined for each class/group of subtours.

3) GENERAL APPLICABILITY OF THE MATHEMATICAL MODEL (15) OF THE NEURO-PROCESSOR TO ANY GRAPH NETWORK

For any graph of known topology (architecture) the fundamental parameters A, A^T, A^+, A^{+T}, D, D^T, S and S^T of (15) are easily obtained. The procedures leading to the determination of the fundamental parameters have been explained above. These fundamental parameters are essential as they are used to solve the model of the neuro-processor TSP solver in (15). This strong point of the neuro-processor model in (15) underscores the flexibility and also the general applicability of (15). Further the parameters/coefficients of (15) are expressed in matrix-form and, this witnesses the scalability potential of (15). In (15), the matrices, D, and S are expressed in general forms applicable to any graph as shown in (15h). The dimensions are defined as follows: $\dim(A)=M\times N$, $\dim(D)=\left(\frac{N}{2}\right)\times N$, $\dim(S)=\aleph\times N$.

In the incidence matrix A, rows correspond to nodes (N_1, $N_2,\ldots N_M$) and columns correspond to edges ($x_1, x_2,\ldots x_N$). Therefore M and N are respectively the magnitude and size of the graph. In matrix D, the rows correspond to pairs of parallel edges (D_1, $D_2,\ldots D_{N/2}$) and columns correspond to edges (x_1, $x_2,\ldots x_N$). Therefore $N/2$ and N are respectively the number of pairs of parallel edges and size of the graph. In matrix S (defined for each class/group of subtours), rows correspond to edges forming a given class/group of subtours (S_1, $S_2,\ldots S_k$) and columns correspond to edges ($x_1, x_2,\ldots x_N$). Therefore $\aleph$ and N are respectively the total number of subtours in a given class/group of subtours and size of the graph.

The connectivity of each edge is expressed in the matrix A by a value chosen in the set $\{-1, 0, 1\}$. These values express a specific connectivity between nodes through edges as follows:

1) The values "-1" and "1" in the first column of A (see rows 1 and 2) show that nodes N_1 and N_2 are connected through the edge x_1, which is in-going to N_1 and out- going from N_2. This connectivity is depicted as $N_2 \xRightarrow{x_1} N_1$.

2) The values "1" and " -1" in the second column of A (see rows 1 and 2) show that the edge x_2 connects N_1 and N_2. The edge x_2 is in-going to N_2 and out-going from N_1. The connectivity of N_1 with N_2 through x_2 is $N_1 \overset{x_2}{\Longrightarrow} N_2$.
3) According to 1) and 2), x_1 and x_2 are parallel edges.
4) The elements "0" in the third row of A ((see columns 1 and 2) show that x_1 and x_2 are not connected to node N_3.

The description provided through the four points above is applicable to any type of graph under investigation in order to obtain the incidence matrix of the graph denoted A in (15h). The derivation of D and S has been already explained above.

IV. STABILITY ANALYSIS AND CONVERGENCE OF THE NEURO-PROCESSOR BASED TSP SOLVER

Our aim in this section is to investigate the local stability of the mathematical model of the neuro-processor tsp solver in (15). The motivation for investigating the local stability is to derive fundamental analytical expressions to ensure (or guarantee) the convergence of the neuro-processor to the exact/true TSP solution. The analytical expressions derived are very important for a systematic control and monitoring of both stability and convergence properties of the neuro-processor.

We now consider the local stability analysis namely the stability of equilibrium points also called fixed points. The equilibrium points $E\left(\vec{x}_e, \vec{\lambda}_e, \vec{\varphi}_e, \vec{\gamma}_e, \vec{\gamma}_e^{*}, \vec{\omega}_e, \vec{\mu}_{le}\right)$

$$A = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & \cdots & x_i & \cdot & \cdot & x_N & \\ -1 & 1 & 0 & 0 & \cdots & 1 & \cdot & \cdot & -1 & N_1 \\ 1 & -1 & 1 & -1 & \cdots & 0 & \cdot & \cdot & 0 & N_2 \\ 0 & 0 & -1 & 1 & \cdots & 1 & \cdot & \cdot & 1 & N_3 \\ \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdot \\ 1 & 0 & -1 & \cdot & \cdots & 1 & \cdot & 0 & \cdot & N_j \\ \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdots & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & -1 & 1 & \cdots & \cdots & -1 & \cdot & \cdot & 1 & N_M \end{bmatrix}_{v\times k}$$

$$D = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & x_{N-1} & x_N & \\ 1 & 1 & 0 & 0 & \cdot & \cdot & 0 & 0 & 0 & 0 & 0 & 0 & D_1 \\ 0 & 0 & 1 & 1 & \cdot & \cdot & 0 & 0 & 0 & 0 & 0 & 0 & D_2 \\ 0 & 0 & 0 & 0 & \cdot & \cdot & 0 & 0 & 0 & 0 & 0 & 0 & D_3 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \vdots \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \vdots \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \vdots \\ 0 & 0 & 0 & 0 & \cdot & \cdot & 1 & 1 & 0 & 0 & 0 & 0 & \vdots \\ 0 & 0 & 0 & 0 & \cdot & \cdot & 0 & 0 & 1 & 1 & 0 & 0 & \vdots \\ 0 & 0 & 0 & 0 & \cdot & \cdot & 0 & 0 & 0 & 0 & 1 & 1 & D_{\frac{N}{2}} \end{bmatrix}$$

$$S = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} & x_{11} & x_{12} & x_{13} & x_{14} & x_{15} & x_{16} & x_{17} & x_{18} & x_{19} & x_{20} & x_{21} & x_{22} & x_{23} & \cdots & x_N & \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & \vdots & s_1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & \vdots & s_2 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & \vdots & s_3 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots & \vdots & s_4 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \cdots & \vdots & s_5 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \cdots & \vdots & s_6 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \cdots & \vdots & s_7 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & S_{40} \\ \vdots & & \ddots & & \ddots & & \ddots & & \ddots & & \ddots & & \ddots & & \ddots & & \ddots & & \ddots & & \ddots & & \ddots & & & \vdots \\ & \cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots & & \cdots & & S_K \end{bmatrix} \quad (15h)$$

of (15) are obtained as solutions of (16) as follows:

$$\begin{aligned}
&A^T\vec{\lambda}_e + A^{+^T}\vec{\varphi}_e \\
&\quad + (2\vec{x}-1)\circ\vec{\gamma}_e + \left(4\vec{x}_e^3 - 6\vec{x}_e^2 + 2\vec{x}_e\right)\circ\vec{\gamma}_e^* \\
&\quad + \left(2D^TD\vec{x}_e - 1\right)\circ D^T\vec{\omega}_e + \vec{U} \\
&\quad + \sum\nolimits_{l=1}^{n}\left[\left(S^TS\vec{x}_e\right)^\circ\left(S^TS\vec{x}_e - 1\right)^\circ\right. \\
&\quad \left.\ldots^\circ\left(S^TS\vec{x}_e - \mathbb{N}+1\right)\right] \\
&\quad \times\left[\left(S^TS\vec{x}_e\right)^{-1} + \left(S^TS\vec{x}_e - 1\right)^{-1}\right. \\
&\quad \left. + \ldots + \left(S^TS\vec{x}_e - \mathbb{N}+1\right)^{-1}\right]\circ S^T\vec{\mu}_{le}\Big\} = 0 \qquad (16a)
\end{aligned}$$

$$A\vec{x}_e = 0 \qquad (16b)$$

$$A^+\vec{x}_e - 2 = 0 \qquad (16c)$$

$$\vec{x}_e\circ(\vec{x}_e - 1) = 0 \qquad (16d)$$

$$\left(\vec{x}_e^4 - 2\vec{x}_e^3 + \vec{x}_e^2\right) = 0 \qquad (16e)$$

$$D\vec{x}_e\circ(D\vec{x}_e - 1) = 0 \qquad (16f)$$

$$\left[S\vec{x}_e^\circ\left(S\vec{x}_e - 1\right)^\circ\ldots^\circ\left(S\vec{x}_e - \mathbb{N}+1\right)\right] = 0 \qquad (16g)$$

According to the equilibriums $P_1(x = 0)$ and $P_2(x = 1)$ depicted/shown in Figure 2, the general solutions of (16) are split up into two sets as it is clearly expressed in (17) and (18).

$$\begin{cases}
\vec{x}_{e1} = 0 \\
A^T\vec{\lambda}_{e1} + A^{+^T}\vec{\varphi}_{e1} - \vec{\gamma}_{e1} - D^T\vec{\omega}_{e1} + \vec{U} + \eta_1 S^T\vec{\mu}_{le1} = 0 \\
\eta_1 = \sum\nolimits_{l=1}^{n}\left(\left[\prod\nolimits_{i=0}^{\mathbb{N}-1}\left(S^TS\vec{x}_{e1} - i\right)\right]\right. \\
\qquad \left.\times\left[\sum\nolimits_{i=0}^{\mathbb{N}-1}\left(S^TS\vec{x}_{e1} - i\right)^{-1}\right]\right)
\end{cases} \qquad (17)$$

$$\begin{cases}
\vec{x}_{e2} = 1 \\
A^T\vec{\lambda}_{e2} + A^{+^T}\vec{\varphi}_{e2} + \vec{\gamma}_{e2} + D^T\vec{\omega}_{e2} + \vec{U} + \eta_2 S^T\vec{\mu}_{le2} = 0 \\
\eta_2 = \sum\nolimits_{l=1}^{n}\left(\left[\prod\nolimits_{i=0}^{\mathbb{N}-1}\left(S^TS\vec{x}_{e2} - i\right)\right]\right. \\
\left.\times\left[\sum\nolimits_{i=0}^{\mathbb{N}-1}\left(S^TS\vec{x}_{e2} - i\right)^{-1}\right]\right)
\end{cases} \qquad (18)$$

According to (17) and (18) the equilibrium points $E\left(\vec{x}_e, \vec{\lambda}_e, \vec{\varphi}_e, \vec{\gamma}_e, \vec{\gamma}_e^*, \vec{\omega}_e, \vec{\mu}_{le}\right)$ are split up into two sets. The first set defined by $E\left(\vec{x}_{e1}, \vec{\lambda}_{e1}, \vec{\varphi}_{e1}, \vec{\gamma}_{e1}, \vec{\gamma}_{e1}^*, \vec{\omega}_{e1}, \vec{\mu}_{le1}\right)$ is obtained as solution of (17). This set encompasses/comprises components, which do not belong to the TSP solution. The second set $E\left(\vec{x}_{e2}, \vec{\lambda}_{e2}, \vec{\varphi}_{e2}, \vec{\gamma}_{e2}, \vec{\gamma}_{e2}^*, \vec{\omega}_{e2}, \vec{\mu}_{le2}\right)$ is obtained as solution of (18). This set encompasses/comprises components, which do belong to the TSP solution.

It is worth mentioning that the results in (17) and (18) can be easily verified and validated through the direct numerical simulation of (15). Indeed, our various numerical simulations of system (15) have revealed that when (15) converges to the exact TSP solution, the results $\left[\vec{x}, \vec{\lambda}, \vec{\varphi}, \vec{\gamma}, \vec{\gamma}^*, \vec{\omega}, \vec{\mu}_l\right]^T$ obtained as solution of (15) are also solutions of (17) and (18). In fact, it has been observed/obtained numerically that the solutions $\left[\vec{x}, \vec{\lambda}, \vec{\varphi}, \vec{\gamma}, \vec{\gamma}^*, \vec{\omega}, \vec{\mu}_l\right]^T$ of (15) when $\vec{x}$ converges to "0" are equal to $E\left(\vec{x}_{e1}, \vec{\lambda}_{e1}, \vec{\varphi}_{e1}, \vec{\gamma}_{e1}, \vec{\gamma}_{e1}^*, \vec{\omega}_{e1}, \vec{\mu}_{e1}\right)$ and the solutions $\left[\vec{x}, \vec{\lambda}, \vec{\varphi}, \vec{\gamma}, \vec{\gamma}^*, \vec{\omega}, \vec{\mu}_l\right]^T$ of (15) when $\vec{x}$ converges to "1" are equal to $E\left(\vec{x}_{e2}, \vec{\lambda}_{e2}, \vec{\varphi}_{e2}, \vec{\gamma}_{e2}, \vec{\gamma}_{e2}^*, \vec{\omega}_{e2}, \vec{\mu}_{e2}\right)$.

We now perturb system (15) around the equilibrium points $E\left(\vec{x}_e, \vec{\lambda}_e, \vec{\varphi}_e, \vec{\gamma}_e, \vec{\omega}_e, \vec{\mu}_{le}\right)$ as follows:

$$\begin{cases}
\vec{x} = \vec{x}_e + \vec{x}_0 \\
\vec{\lambda} = \vec{\lambda}_e + \vec{\lambda}_0;\ \vec{\varphi} = \vec{\varphi}_e + \vec{\varphi}_0;\ \vec{\gamma} = \vec{\gamma}_e + \vec{\gamma}_0 \\
\vec{\gamma}^* = \vec{\gamma}_e^* + \vec{\gamma}_0^*;\ \vec{\omega} = \vec{\omega}_e + \vec{\omega}_0;\ \vec{\mu} = \vec{\mu}_{le} + \vec{\mu}_{l0}
\end{cases} \qquad (19)$$

where $\vec{x}_0$, $\vec{\lambda}_0$, $\vec{\varphi}_0$, $\vec{\gamma}_0$, $\vec{\gamma}_0^*$, $\vec{\omega}_0$ and $\vec{\mu}_{l0}$ are small perturbations expressed as $\vec{x}_0 \to 0, \vec{\lambda}_0 \to 0, \vec{\varphi}_0 \to 0, \vec{\gamma}_0 \to 0, \vec{\gamma}_0^* \to 0 \vec{\omega}_0 \to 0$, and $\vec{\mu}_{l0} \to 0$. Substituting (19) into (15) and ignoring high-order perturbations (e.g. $\vec{x}_0\vec{\lambda}_0 = 0$, $\vec{x}_0\vec{\omega}_0 = 0$, $\vec{x}_0^2 = 0$, $\vec{x}_0^3 = 0$, $(S\vec{x}_0)^2$, $\left(S^TS\vec{x}_0\right)^2$, and $(S\vec{\mu}_{l0})\left(S^TS\vec{x}_0\right)$, etc.) leads to the variational equation expressed in compact (or matrix) form as illustrated in (20a).

$$\frac{d}{dt}\left[\vec{x}_0, \vec{\lambda}_0, \vec{\varphi}_0, \vec{\gamma}_0, \vec{\gamma}_0^*, \vec{\omega}_0, \vec{\mu}_{l0}\right]^T = [M_J]\left[\vec{x}_0, \vec{\lambda}_0, \vec{\varphi}_0, \vec{\gamma}_0, \vec{\gamma}_0^*, \vec{\omega}_0, \vec{\mu}_{l0}\right]^T \qquad (20a)$$

The variational matrix M_J (also called Jacobian matrix) in Eq. (20a) is defined in Eq. (20b), as shown at the next page. In Eq. (20b), the quantities $F(\vec{x}_e)$, $G(\vec{x}_e)$, and $H(\vec{x}_e)$ are defined as follows:

$$\begin{aligned}
&F(\vec{x}_e) \\
&\ = \left[\prod\nolimits_{i=0}^{\mathbb{N}-1}\left(S^TS\vec{x}_e - i\right)\right] \\
&\quad \times\left[\sum\nolimits_{i=0}^{\mathbb{N}-1}\sum\nolimits_{j=0}^{\mathbb{N}-1}\left(S^TS\vec{x}_e - i\right)^{-1}\left(S^TS\vec{x}_e - j\right)^{-1}\right] \\
&\quad \times\left(S^T\vec{\mu}_e\right)S^TS(i \neq j) \qquad (20c)
\end{aligned}$$

$$\begin{aligned}
&G(\vec{x}_e) \\
&\ = \left[\prod\nolimits_{i=0}^{\mathbb{N}-1}\left(S^TS\vec{x}_e - i\right)\right]\times\left[\sum\nolimits_{i=0}^{\mathbb{N}-1}\left(S^TS\vec{x}_e - i\right)^{-1}\right]S^T \qquad (20d)
\end{aligned}$$

$$\begin{aligned}
&H(\vec{x}_e) \\
&\ = \left[\prod\nolimits_{i=0}^{\mathbb{N}-1}(S\vec{x}_e - i)\right]\times\left[\sum\nolimits_{i=0}^{\mathbb{N}-1}(S\vec{x}_e - i)^{-1}\right]S \qquad (20e)
\end{aligned}$$

According to (17) and (18) the matrix M_J can be expressed into the simplified form (21), as shown at the next page.

$$M_J = \begin{bmatrix} -2\alpha\vec{\gamma}_e - \alpha(12\vec{x}_e^2 - 12\vec{x}_e + 2)\vec{\gamma}_e^* - 2\alpha(D^T\vec{\omega}_e)D^T D - \alpha F(\vec{x}_e) & -\alpha A^T & -\alpha A^{+T} & -\alpha\,(2\vec{x}_e - 1) & -\alpha(4\vec{x}_e^3 - 6\vec{x}_e^2 + 2\vec{x}_e) & -\alpha\left(2D^T D\vec{x}_e - 1\right)D^T & -\alpha \mathrm{G}(\vec{x}_e) \\ \beta_1 A & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_2 A^+ & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_3(2\vec{x}_e - 1) & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_3^*(4\vec{x}_e^3 - 6\vec{x}_e^2 + 2\vec{x}_e) & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_4\,(2D\vec{x}_e - 1)\,D & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_{5,l} H(\vec{x}_e) & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{20b}$$

$$M_J = \begin{bmatrix} -2\alpha\vec{\gamma}_e - 2\alpha\vec{\gamma}_e^* - 2\alpha(D^T\vec{\omega}_e)D^T D - \alpha F(\vec{x}_e) & -\alpha A^T & -\alpha A^{+T} & -\alpha\,(2\vec{x}_e - 1) & 0 & -\alpha\left(2D^T D\vec{x}_e - 1\right)D^T & -\alpha \mathrm{G}(\vec{x}_e) \\ \beta_1 A & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_2 A^+ & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_3(2\vec{x}_e - 1) & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_4\,(2D\vec{x}_e - 1)\,D & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_{5,l} H(\vec{x}_e) & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{21}$$

$$M_{J1} = \begin{bmatrix} -2\alpha\vec{\gamma}_{e1} - 2\alpha\vec{\gamma}_{e1}^* - 2\alpha(D^T\vec{\omega}_{e1})D^T D - \alpha\eta_5(S^T\vec{\mu}_{e1})S^T S & -\alpha A^T & -\alpha A^{+T} & \alpha & 0 & \alpha D^T & -\alpha\eta_1 S^T \\ \beta_1 A & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_2 A^+ & 0 & 0 & 0 & 0 & 0 & 0 \\ -\beta_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\beta_4 D & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_{5,l}\eta_3 S & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{22}$$

$$M_{J2} = \begin{bmatrix} -2\alpha\vec{\gamma}_{e2} - 2\alpha\vec{\gamma}_{e2}^* - 2\alpha(D^T\vec{\omega}_{e2})D^T D - \alpha\eta_6(S^T\vec{\mu}_{e2})S^T S & -\alpha A^T & -\alpha A^{+T} & -\alpha & 0 & -\alpha D^T & -\alpha\eta_2 S^T \\ \beta_1 A & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_2 A^+ & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_4 D & 0 & 0 & 0 & 0 & 0 & 0 \\ \beta_{5,l}\eta_4 S & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{23}$$

$$\text{Where}: \ \eta_3 = \left[\prod_{i=0}^{\mathbb{N}-1}(S\vec{x}_{e1} - i)\right] \times \left[\sum_{i=0}^{\mathbb{N}-1}(S\vec{x}_{e1} - i)^{-1}\right], \quad \eta_4 = \left[\prod_{i=0}^{\mathbb{N}-1}(S\vec{x}_{e2} - i)\right] \times \left[\sum_{i=0}^{\mathbb{N}-1}(S\vec{x}_{e2} - i)^{-1}\right],$$

$$\eta_5 = \left[\prod_{i=0}^{\mathbb{N}-1}\left(S^T S\vec{x}_{e1} - i\right)\right] \times \left[\sum_{i=0}^{\mathbb{N}-1}\sum_{j=0}^{\mathbb{N}-1}\left(S^T S\vec{x}_{e1} - i\right)^{-1}\left(S^T S\vec{x}_{e1} - j\right)^{-1}\right],$$

$$\eta_6 = \left[\prod_{i=0}^{\mathbb{N}-1}\left(S^T S\vec{x}_{e2} - i\right)\right] \times \left[\sum_{i=0}^{\mathbb{N}-1}\sum_{j=0}^{\mathbb{N}-1}\left(S^T S\vec{x}_{e2} - i\right)^{-1}\left(S^T S\vec{x}_{e2} - j\right)^{-1}\right]$$

Considering (21), the Jacobian matrices M_{J1} and M_{J2} representing the first set (see (17)) and the second set (see (18)) are expressed in (22) and (23), as shown at the previous page, respectively.

The quantities η_1 and η_2 are defined in (17) - (18). All η_i $(i = 1,2,3,4,5,6)$ are positive integers calculated at equilibrium. Thus, the Jacobian matrix M_{J1} corresponds to equilibrium points $E\left(\vec{x}_{e1}, \vec{\lambda}_{e1}, \vec{\varphi}_{e1}, \vec{\gamma}_{e1}, \vec{\omega}_{e1}, \vec{\mu}_{le1}\right)$ in the first set obtained as solution of (17). Similarly, M_{J2} corresponds to equilibrium points $E\left(\vec{x}_{e2}, \vec{\lambda}_{e2}, \vec{\varphi}_{e2}, \vec{\gamma}_{e2}, \vec{\omega}_{e2}, \vec{\mu}_{le2}\right)$ in the second set obtained as solution of (18). Using (22) the characteristic equation also called eigenvalues equation is expressed in (24) from which the stability of equilibrium points $E\left(\vec{x}_{e1}, \vec{\lambda}_{e1}, \vec{\varphi}_{e1}, \vec{\gamma}_{e1}, \vec{\omega}_{e1}, \vec{\mu}_{le1}\right)$ is analyzed. Similarly, (23) is used to express (or define) the characteristic equation in (25). This equation is further used to analyze the stability of equilibrium/fixed points $E\left(\vec{x}_{e2}, \vec{\lambda}_{e2}, \vec{\varphi}_{e2}, \vec{\gamma}_{e2}, \vec{\omega}_{e2}, \vec{\mu}_{le2}\right)$.

$$det\left[M_{J1} - \lambda I_{\mathrm{d}}\right] = 0 \tag{24}$$

$$det\left[M_{J2} - \lambda I_{\mathrm{d}}\right] = 0 \tag{25}$$

The quantity λ represents the eigenvalue from which the stability of equilibrium/fixed points is deduced. Specifically, an equilibrium point is stable if all real parts of λ are negative. Otherwise, the equilibrium/fixed point is unstable [49].

An expansion of (24) leads to the characteristic equation in (26). This is a quadratic algebraic equation.

$$\begin{aligned}\lambda^2 &+ 2\alpha\left[\vec{\gamma}_{e1} + (D^T\vec{\omega}_{e1})D^TD + \frac{1}{2}\eta_5(S^T\vec{\mu}_{le1})S^TS\right]\lambda \\ &+ \alpha\left[\beta_1A^TA + \beta_2A^{+T}A^{+} + \beta_4D^TD + \beta_{5,l}\eta_1\eta_3S^TS\right] = 0\end{aligned} \tag{26}$$

Similarly, an expansion of (25) leads to the characteristic equation in (27).

$$\begin{aligned}\lambda^2 &+ 2\alpha\left[\vec{\gamma}_{e2} + (D^T\vec{\omega}_{e2})D^TD + \frac{1}{2}\eta_6(S^T\vec{\mu}_{le2})S^TS\right]\lambda \\ &+ \alpha\left[\beta_1A^TA + \beta_2A^{+T}A^{+} + \beta_4D^TD + \beta_{5,l}\eta_2\eta_4S^TS\right] = 0\end{aligned} \tag{27}$$

The solutions of (26) are expressed in (28).

$$\begin{aligned}\lambda_1 &= -\alpha\left[\vec{\gamma}^*_{e1} + \vec{\gamma}_{e1} + (D^T\vec{\omega}_{e1})D^TD + \frac{1}{2}\eta_5(S^T\vec{\mu}_{le1})S^TS\right] \\ &+ \sqrt{\begin{array}{l}\alpha^2\left[\vec{\gamma}^*_{e1} + \vec{\gamma}_{e1} + (D^T\vec{\omega}_{e1})D^TD + \frac{1}{2}\eta_5(S^T\vec{\mu}_{le1})S^TS\right]^2 \\ -\alpha\left[\beta_1A^TA + \beta_2A^{+T}A^{+} + \beta_4D^TD + \beta_{5,l}\eta_1\eta_3S^TS\right]\end{array}}\end{aligned} \tag{28a}$$

$$\begin{aligned}\lambda_2 &= -\alpha\left[\vec{\gamma}^*_{e1} + \vec{\gamma}_{e1} + (D^T\vec{\omega}_{e1})D^TD + \frac{1}{2}\eta_5(S^T\vec{\mu}_{le1})S^TS\right] \\ &- \sqrt{\begin{array}{l}\alpha^2\left[\vec{\gamma}^*_{e1} + \vec{\gamma}_{e1} + (D^T\vec{\omega}_{e1})D^TD + \frac{1}{2}\eta_5(S^T\vec{\mu}_{le1})S^TS\right]^2 \\ -\alpha\left[\beta_1A^TA + \beta_2A^{+T}A^{+} + \beta_4D^TD + \beta_{5,l}\eta_1\eta_3S^TS\right]\end{array}}\end{aligned} \tag{28b}$$

An important comment, which is worth mentioning, is that in (28), the quantities $\vec{\omega}_{e1}$, $\vec{\gamma}_{e1}$, $\vec{\gamma}^*_{e1}$, and $\vec{\mu}_{le1}$ are known vectors of real numbers obtained as solutions of (17). The equilibrium points $E\left(\vec{x}_{e1}, \vec{\lambda}_{e1}, \vec{\varphi}_{e1}, \vec{\gamma}_{e1}, \vec{\omega}_{e1}\right)$ are stable if all real parts of the eigenvalues λ_1 and λ_2 in (28) are negative [49]. As already mentioned, (see in Section III. b) the quantities α and β_i represent the step sizes for updating decision- and multiplier-variables respectively. Also, these quantities are positive real numbers. Thus, a condition to obtain negative values for all real parts of λ_1 and λ_2 in (28) can be expressed by (29) $(\alpha > 0;\ \beta_i > 0)$. The condition (29) ensures the stability of $E\left(\vec{x}_{e1}, \vec{\lambda}_{e1}, \vec{\varphi}_{e1}, \vec{\gamma}_{e1}, \vec{\omega}_{e1}\right)$.

$$\begin{cases}\left[\vec{\gamma}^*_{e1} + \vec{\gamma}_{e1} + (D^T\vec{\omega}_{e1})D^TD + \frac{1}{2}\eta_5(S^T\vec{\mu}_{le1})S^TS\right] > 0 \\ \alpha\left[\vec{\gamma}^*_{e1} + \vec{\gamma}_{e1} + (D^T\vec{\omega}_{e1})D^TD + \frac{1}{2}\eta_5(S^T\vec{\mu}_{le1})S^TS\right]^2 \\ -\left[\beta_1A^TA + \beta_2A^{+T}A^{+} + \beta_4D^TD + \beta_{5,l}\eta_1\eta_3S^TS\right] < 0\end{cases} \tag{29}$$

Similarly, the solutions of Eq. (27) are expressed in (30).

$$\begin{aligned}\lambda_1 &= -\alpha\left[\vec{\gamma}^*_{e2} + \vec{\gamma}_{e2} + (D^T\vec{\omega}_{e2})D^TD + \frac{1}{2}\eta_6(S^T\vec{\mu}_{le2})S^TS\right] \\ &+ \sqrt{\begin{array}{l}\alpha^2\left[\vec{\gamma}^*_{e2} + \vec{\gamma}_{e2} + (D^T\vec{\omega}_{e2})D^TD + \frac{1}{2}\eta_6(S^T\vec{\mu}_{le2})S^TS\right]^2 \\ -\alpha\left[\beta_1A^TA + \beta_2A^{+T}A^{+} + \beta_4D^TD + \beta_{5,l}\eta_2\eta_4S^TS\right]\end{array}}\end{aligned} \tag{30a}$$

$$\begin{aligned}\lambda_2 &= -\alpha\left[\vec{\gamma}^*_{e2} + \vec{\gamma}_{e2} + (D^T\vec{\omega}_{e2})D^TD + \frac{1}{2}\eta_6(S^T\vec{\mu}_{le2})S^TS\right] \\ &- \sqrt{\begin{array}{l}\alpha^2\left[\vec{\gamma}^*_{e2} + \vec{\gamma}_{e2} + (D^T\vec{\omega}_{e2})D^TD + \frac{1}{2}\eta_6(S^T\vec{\mu}_{le2})S^TS\right]^2 \\ -\alpha\left[\beta_1A^TA + \beta_2A^{+T}A^{+} + \beta_4D^TD + \beta_{5,l}\eta_2\eta_4S^TS\right]\end{array}}\end{aligned} \tag{30b}$$

According to (30), one can derive (31) as a fundamental condition to obtain negative values for all real parts of λ_1 and λ_2 $(\alpha > 0;\ \beta_i > 0)$. Thus, condition (31) ensures the stability of equilibrium points $E\left(\vec{x}_{e2}, \vec{\lambda}_{e2}, \vec{\varphi}_{e2}, \vec{\gamma}_{e2}, \vec{\omega}_{e2}\right)$.

$$\begin{cases}\left[\vec{\gamma}^*_{e2} + \vec{\gamma}_{e2} + (D^T\vec{\omega}_{e2})D^TD + \frac{1}{2}\eta_6(S^T\vec{\mu}_{e2})S^TS\right] > 0 \\ \alpha\left[\vec{\gamma}^*_{e2} + \vec{\gamma}_{e2} + (D^T\vec{\omega}_{e2})D^TD + \frac{1}{2}\eta_6(S^T\vec{\mu}_{le2})S^TS\right]^2 \\ -\left[\beta_1A^TA + \beta_2A^{+T}A^{+} + \beta_4D^TD + \beta_{5,l}\eta_2\eta_4S^TS\right] < 0\end{cases} \tag{31}$$

Comments on analytical conditions (29) and (31):

1) The analytical conditions (29) and (31) correspond both to the properties of "positive definiteness" and "negative definiteness". When these properties are simultaneously fulfilled (according to (29) and (31)), the neuro-processor model in (15) always converges to the true/exact TSP solution. Therefore (29) and (31) are key conditions used in this paper to control and ensure/guarantee convergence of the neuro-processor to the exact/true TSP solution/tour
2) For a given graph under investigation, the matrices A, D, and S are obtained and are further used to check/verify the analytical conditions (29) and (31) through numerical simulations. The aim of the numerical simulation here is to obtain several sets of parameters ($\alpha, \beta_1, \beta_2, \beta_4, \beta_{5,l}$) that simultaneously fulfil the conditions in (29) and (31). Using the sets of parameters ($\alpha, \beta_1, \beta_2, \beta_4, \beta_{5,l}$) obtained in the frame of the numerical solving of the neuro-processor model in (15), various numerical simulations are performed to obtain the solution $\vec{x}$ from which the edges belonging to the exact/true TSP tour are depicted.
3) Let's note that the traditional/classical methods, concepts and algorithms for TSP solving (e.g. GA [30], (SA) [28], EA [14], ACO [36], PSO [41], ANN [29], etc.) do not provide such a systematic analytical framework to ensure/guarantee convergence to the exact TSP solution. Thus, the analytical expressions established in (29) and (31) appear as a contribution which could significantly enrich the state-of-the-art regarding TSP solving in dynamically reconfigurable graph-networks.
4) The conditions (29) and (31), which are derived to ensure/guarantee convergence to the exact TSP solution depend on: (a) the step size α of decision variables (also called decision neurons); (b) the step size of multiplier variables/neurons β_1, β_2, β_4 and $\beta_{5,l}$; (c) the incidence matrix A of graph G; (d) the matrix D expressing parallel edges in G; (e) the matrix S describing subtours of G.

The analytical results obtained in Sections III and IV are validated in the next section (Section V) through various numerical simulations, which are carried out to solve TSP for two recently published case-study examples (see [11] and [26]), this for both validation and benchmarking purposes.

V. NUMERICAL SIMULATIONS AND PROOF OF CONCEPTS OF THE NEURO-PROCESSOR DEVELOPED

A. APPLICATION EXAMPLE 1A: COMPUTATION OF "THE TSP TOUR" IN A CITY GRAPH (RESULTS PUBLISHED IN [11]): BENCHMARKING WITH THE NEURO-PROCESSOR DEVELOPED

The application example published in [11] computes the tsp tour in the city graph with distance weights in figure 3. For the sake of simplifying notations in figure 3, the weights values

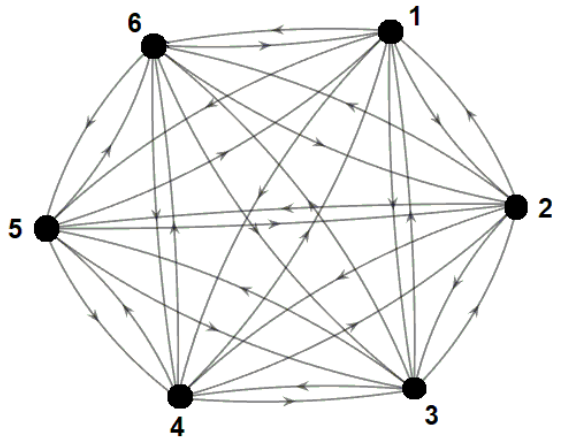


FIGURE 3. A city graph with distance weights: Case study published in [11]. Nodes are pair-wisely connected by bi-directional edges. The optimal TSP tour is obtained as numerical solution of (15).

of all edges in figure 3 are defined as follows: The cost of the edge $C_{1\to2}$ connecting node 1 and node 2 is denoted by c_1 ($c_{1\to2} = c_1$), $c_{2\to1}$ is denoted by c_2 ($c_{2\to1} = c_2$), $c_{1\to3} = c_3$, $c_{3\to1} = c_4$, $c_{1\to4} = c_5$, $c_{4\to1} = c_6$, $c_{1\to5} = c_7$, $c_{5\to1} = c_8$, $c_{1\to6} = c_9$, $c_{6\to1} = c_{10}$, $c_{2\to3} = c_{11}$, $c_{3\to2} = c_{12}$, $c_{2\to4} = c_{13}$, $c_{4\to2} = c_{14}$, $c_{2\to5} = c_{15}$, $c_{5\to2} = c_{16}$, $c_{2\to6} = c_{17}$, $c_{6\to2} = c_{18}$, $c_{3\to4} = c_{19}$, $c_{4\to3} = c_{20}$, $c_{3\to5} = c_{21}$, $c_{5\to3} = c_{22}$, $c_{3\to6} = c_{23}$, $c_{6\to3} = c_{24}$, $c_{4\to5} = c_{25}$, $c_{5\to4} = c_{26}$, $c_{4\to6} = c_{27}$, $c_{6\to4} = c_{28}$, $c_{5\to6} = c_{29}$, $c_{6\to5} = c_{30}$. The state of connectivity $\vec{x}$ of edges in Figure 3 is expressed through components x_i defined as follows: the state of edge $x_{1\to2}$ connecting nodes 1 and 2 is denoted by x_1 ($x_{1\to2} = x_1$), $x_{2\to1}$ connects nodes 1 and 2 (in reverse direction) and is denoted by x_2 ($x_{2\to1} = x_2$). Thus x_1 and x_2 are parallel edges. Similarly, $x_{1\to3} = x_3$, $x_{3\to1} = x_4$, $x_{1\to4} = x_5$, $x_{4\to1} = x_6$, $x_{1\to5} = x_7, x_{5\to1} = x_8, x_{1\to6} = x_9, x_{6\to1} = x_{10}, x_{2\to3} = x_{11}$, $x_{3\to2} = x_{12}$, $x_{2\to4} = x_{13}$, $x_{4\to2} = x_{14}$, $x_{2\to5} = x_{15}$, $x_{5\to2} = x_{16}$, $x_{2\to6} = x_{17}$, $x_{6\to2} = x_{18}$, $x_{3\to4} = x_{19}$, $x_{4\to3} = x_{20}$, $x_{3\to5} = x_{21}$, $x_{5\to3} = x_{22}$, $x_{3\to6} = x_{23}$, $x_{6\to3} = x_{24}$, $x_{4\to5} = x_{25}$, $x_{5\to4} = x_{26}$, $x_{4\to6} = x_{27}$, $x_{6\to4} = x_{28}$, $x_{5\to6} = x_{29}$, $x_{6\to5} = x_{30}$. The belonging of an edge to the TSP- solution is expressed by $x_i = 1$; otherwise $x_i = 0$.

The values of edges used in [11] to compute the TSP tour/solution in Figure 3 are: $c_1 = c_2 = 111.4$; $c_3 = c_4 = 211$; $c_5 = c_6 = 219$; $c_7 = c_8 = 165.7$; $c_9 = c_{10} = 85.4$; $c_{11} = c_{12} = 131.6$; $c_{13} = c_{14} = 185.6$; $c_{15} = c_{16} = 153.4$; $c_{17} = c_{18} = 76.5$; $c_{19} = c_{20} = 94.4$; $c_{21} = c_{22} = 110.1$; $c_{23} = c_{24} = 128$; $c_{25} = c_{26} = 56.8$; $c_{27} = c_{28} = 138.1$; $c_{29} = c_{30} = 90.7$. The values of costs of edges above were used in [11] to compute the optimal TSP tour. As reported in [11], the total cost obtained for the optimal TSP solution/tour is $\tilde{L} = 570.3$.

For the sake of benchmarking, we now use the mathematical model of the neuro-processor in (15) to compute the TSP solution/tour in Figure 3. The 4th order Runge-Kutta algorithm is implemented using the MATLAB version R2019a.

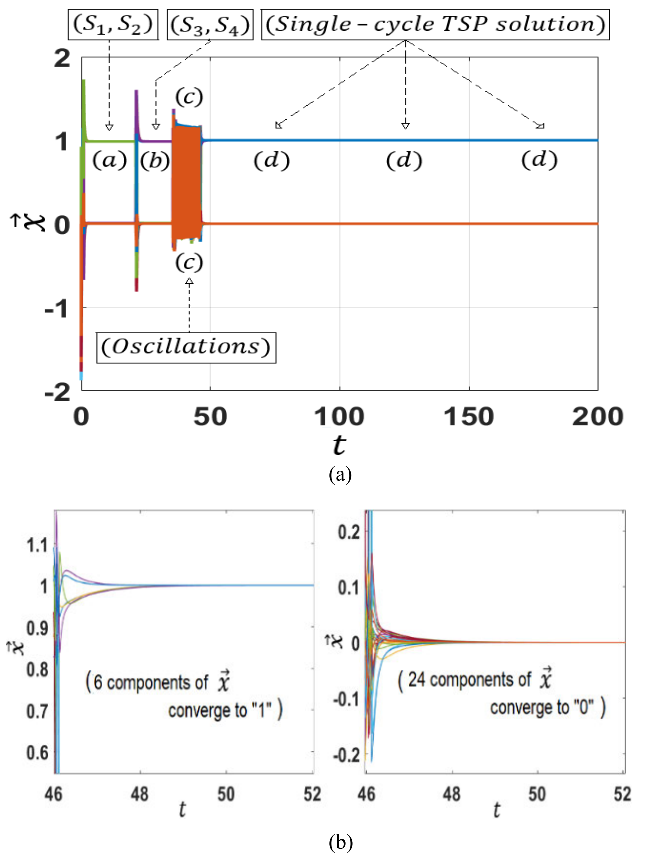


FIGURE 4. (a) Numerical solution $\vec{x}$ of (15). The components of $\vec{x}$ are binary values which are used to detect the TSP solution/tour in Figure 3. (*a*) first TSP solution in form of subtours (S_1, S_2) detected in the range $4 < t < 20$. (*b*) second TSP solution in form of subtours (S_3, S_4) detected in the range $25 < t < 35$. (*c*) Oscillatory solutions $\vec{x}$ detected in the range $36 < t < 46$. (*d*) Final TSP solution in form of a single-cycle detected for all $t > 50$. This latter solution $\vec{x}$ of (15) is stable as it remains unchanged for all $t > 50$. (b) A zooming of the solution $\vec{x}$ in Fig. 4a in the range $46 < t < 52$. This range overlaps the range of oscillatory dynamics ($36 < t < 46$) and the range of permanent dynamics ($t > 50$). The zooming clearly illustrates the convergence of $\vec{x}$ to binary variables "0" and "1". When t increases further ($t > 50$) all solutions $\vec{x}$ merge into two lines: Line 1: $x_1 = x_{10} = x_{11} = x_{19} = x_{25} = x_{29} = 1$ and Line 2: All other "24 components" of $\vec{x}$ are equal zero ($x_i = 0$). This witnesses the binarization of solutions $\vec{x}$ of (15).

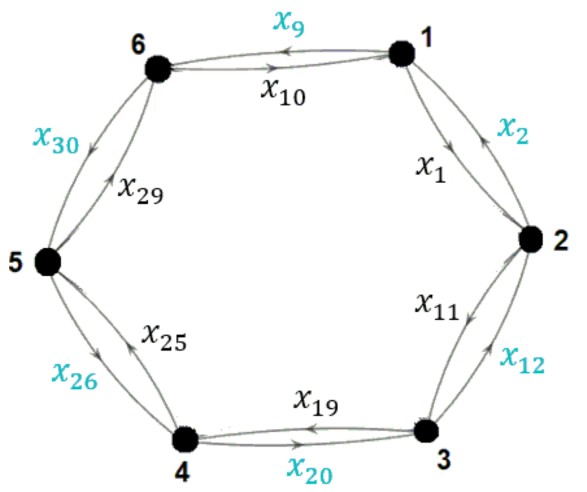


FIGURE 5. TSP tour obtained according to the numerical solution $\vec{x}$ in Fig. 4a. Depending on the initial conditions used to solve (15), the TSP tour detected is $x_1 = x_{10} = x_{11} = x_{19} = x_{25} = x_{29} = 1$ (in one direction) and also $x_1 = x_{10} = x_{11} = x_{19} = x_{25} = x_{29} = 1$ (in reverse direction) as the graph in Fig. 3 is bidirectional. In both directions the total cost of TSP tour is $\tilde{L} = 570.3$.

For the case-study in Figure 3, the matrices A, D, S are defined in (15h). The vector of the costs of edges U is defined by the weights values c_i. Further $\mathbb{N} = 3$ and $\aleph = 40$. Additional coefficients used for the numerical solving of (15) are defined as follows: $\alpha = 5$; $\beta_1 = 100$; $\beta_2 = 100$; $\beta_3 = 180$; $\beta_3^* = 170$; $\beta_4 = 100$; $\beta_5 = 0.01$; $h = 0.000125$ (step size). These values of coefficients are chosen according to stability conditions (29) and (31). The procedure of using (29) and (31) to obtain the values of β_i has already been explained above (*see comments on analytical conditions (29) and (31)*).

Using the values assigned to coefficients, the numerically solving of (15) has led to results depicted/displayed in Figure 4a. Figure 4a shows the temporal evolution of the vector of decision variables $\vec{x} = [x_1, x_2, \ldots, x_{30}]^T$ obtained as numerical solution of (15). As it appears in Figure 4a, the solution $\vec{x}$ of (15) converges to a first TSP solution expressed by subtours (S_1, S_2) in the range $4 < t < 20$. S_1 corresponds to $x_2 = x_9 = x_{18} = 1$ and S_2 corresponds to $x_{20} = x_{21} = x_{26} = 1$. The remaining coefficients of vector $\vec{x}$ are equal zero. The total cost of subtours (S_1, S_2) is $\tilde{L} = 534.6$. As the variable t increases, a second TSP solution is detected and expressed by subtours (S_3, S_4) in the range $25 < t < 35$. S_3 corresponds to $x_1 = x_{10} = x_{17} = 1$ and S_4 corresponds to $x_{19} = x_{22} = x_{25} = 1$. The total cost of subtours (S_3, S_4) is $\tilde{L} = 534.6$. The oscillatory state of the solution $\vec{x}$ of (15) is also depicted in the range $36 < t < 46$. As the variable t increases further, say $t > 50$, the numerical solution $\vec{x}$ of (15) converges to a stable solution characterized by $x_1 = x_{10} = x_{11} = x_{19} = x_{25} = x_{29} = 1$. In this case, the other/remaining coefficients of vector $\vec{x}$ are equal zero. These binary values of $\vec{x}$ are obtained from Figure 4a. The values obtained from Figure 4a correspond to a TSP solution made-up of a single-cycle as depicted in Figure 5. The total cost of the single-cycle TSP tour is $\tilde{L} = 570.3$. This is the optimum TSP-solution with a single-cycle that exists in Figure 3 for the specific values assigned to the costs of edges (see above). It is worth mentioning that identical result is also reported in [11].

A comparison between method 1 (i.e. the one published in [11]) and method 2 (i.e. our neuro-processor concept developed in (15)) is performed in TABLE 1.

B. APPLICATION EXAMPLE 1B: TOUR- STABILITY ANALYSIS OF A "PREVIOUSLY COMPUTED OPTIMAL TSP TOUR" (RESULTS PUBLISHED IN [11]): BENCHMARKING WITH THE NEURO-PROCESSOR DEVELOPED

1) RESULTS OF THE STABILITY ANALYSIS IN [11]

This application example (published in [11]) investigates the stability of a "previously computed traveling salesman tour".

TABLE 1. Benchmarking results for application example 1: comparison of method 1 in [11] with method 2 (the concept developed in Eq. (15)).

Concepts	Exact TSP solution	Convergence	Convergence starts at *t* value?
Concept in [11]	$x_1 \rightarrow x_{11} \rightarrow x_{19} \rightarrow x_{25} \rightarrow x_{29} \rightarrow x_{10}$	Yes (Same exact TSP solution)	**Not reported in [11]**
Our neuro-processor concept	$x_1 \rightarrow x_{11} \rightarrow x_{19} \rightarrow x_{25} \rightarrow x_{29} \rightarrow x_{10}$	Yes (Same exact TSP solution)	$t \approx 50$ (see Fig. 4)

The integer linear programming (ilp) relaxation method is used in [11] for the needed "tour"-stability analysis. The overall procedure for analyzing the stability of a "previously computed traveling salesman tour" conducted in [11] can be summarized through the following steps:

Step 1. Assignment of fixed weights values (or constant costs) to all edges of a graph network and computation of a corresponding optimal TSP solution/tour. This tour is the "target TSP tour" (see TSP tour in Figure 5).

Step 2. Variation of the costs of an arbitrary number of edges (of the graph) through perturbations (denoted Δc_{ij}) and re-computation of TSP ("newly computed TSP tours"). The stability criterion consists of comparing the cost of the "target TSP tour" with the costs of "newly computed TSP tours".

Step 3. Use of the procedure in step 2 to compute regions of "stability" and regions of "instability" according to following statements:

1) If the "previously computed TSP tour", here the "target TSP tour", remains the optimal/best solution (meaning the solution with smallest total cost) compared to a set of "newly computed TSP tours", then the "previously computed TSP tour" is said to be stable.
2) If a "newly computed TSP tour" is discovered with a total cost less than the total cost of the "previously computed TSP tour", then the "previously computed TSP tour" is said to be unstable.

The two key statements above are used in [11] to analyze the stability of the "previously computed TSP tour". The edges subject to perturbations are expressed as $\Delta c_{ij}+c_{ij}$ (Δc_{ij} is the strength/amplitude of perturbation applied on edges c_{ij}). In essence, the stability analysis in [11] consists of perturbing the edges $c_{1\rightarrow 2}$ (edge connecting nodes 1 and 2) and $c_{1\rightarrow 5}$ (edge connecting nodes 1 and 5) in Figure 3. The edges $c_{1\rightarrow 5}$ (with cost c_7), $c_{5\rightarrow 1}$ (with cost c_8), $c_{1\rightarrow 2}$ (with cost c_1), and $c_{2\rightarrow 1}$ (with cost c_2) are subject to perturbations denoted by Δc_{15} and Δc_{12} and their resulting costs are expressed as follows: $c_7 + \Delta c_{15}$, $c_8 + \Delta c_{15}$, $c_1 + \Delta c_{12}$, and $c_2 + \Delta c_{12}$.

Findings/results of the stability analysis published in [11] are depicted in Figure 6. As it appears in Figure 6, the monitoring of strengths of perturbations (of edges) in ranges $-200 \leq \Delta c_{15} \leq 1200$ and $200 \leq \Delta c_{12} \leq 1200$ has led to the computation of regions of stability (see area with grey color in Figure 6) as well as regions of instability (see area with white color in Figure 6 surrounding the area of stability). A specific perturbation $(\Delta c_{12}, \Delta c_{15}) = (+50, -50)$ is also

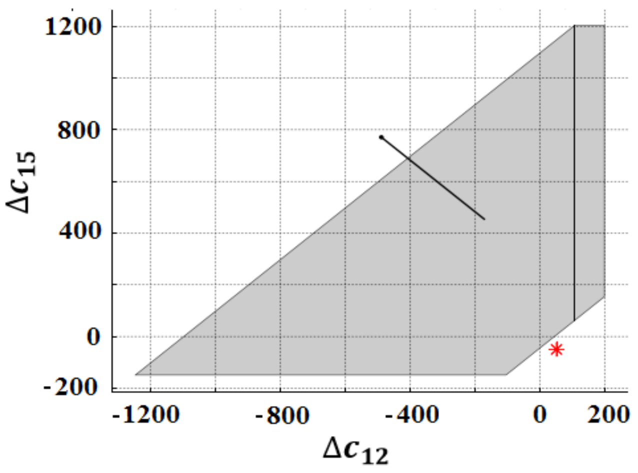


FIGURE 6. Results of the stability analysis (in Figure 3) in terms of Δc_{12} and Δc_{15}. This figure is published in Ref. [11]. Stability region (grey color); a specific point of instability (red star); all comments and concluding remarks formulated on Figure 6 are reported in [11].

considered in [11] and the computation of new TSP solution with edges subject to aforementioned specific perturbations has led to the discovery of a "new best/optimal TSP tour" when compared with the "target TSP tour" computed using $(\Delta c_{12}, \Delta c_{15}) = (+50, 0)$. Thus, the "target TSP tour", here the "previously computed traveling salesman tour", is said to be unstable. The point at which the "previously computed TSP solution/tour" is said to be unstable is materialized by a red star located in Figure 6 with coordinates $(\Delta c_{12}, \Delta c_{15}) = (+50, -50)$.

Overall, as it appears in Figure 6, when the perturbations Δc_{12} and Δc_{15} (applied to costs of edges) increase, the previously computed tour may remain stable (see region in grey color) or may become unstable (see region in white color).

2) BENCHMARKING OF RESULTS IN [11] WITH RESULTS PROVIDED BY THE NEURO-PROCESSOR MODEL IN (15)

The stability analysis is now conducted using the numerical simulation of (15). This is the mathematical model of the neuro-processor based TSP solver developed. In order to facilitate the benchmarking we used exactly same/identical values of parameters as reported in [11].

We now use the neuro-processor model (15) to investigate the stability of the "previously computed traveling salesman tour" as reported in [11] (see Figure 6 published in [11]). Similarly to study in [11], we perturb the costs of edges as follows: c_1 $(c_1+\Delta c_{12})$, c_2 $(c_2+\Delta c_{12})$, c_7 $(c_7+\Delta c_{15})$ and c_8 $(c_8+\Delta c_{15})$. The perturbed edges are used as control parameters between the ranges $1200 \leq \Delta c_{12} \leq 200$ and $-200 \leq \Delta c_{15} \leq 1200$. The main objective here is to demonstrate that the analysis of stability using the neuro-processor in (15) is straightforward/easy and also leads to results of high accuracy, and good robustness. As in previous analysis, the results provided by the neuro-processor are obtained as numerical solutions of (15). The values of parameters A, U, D, S, α, β_1, β_2, β_3, β_3^*, β_4, and β_5 used to solve (15) are identical/same as

TABLE 2. Extension of results published in [11]: results of the stability analysis of the "previously computed traveling salesman tours" using the neuro-processor based concept developed in this paper. Black numbers represent the total costs of the "previously computed traveling salesman tours" while the total cost of the "newly computed traveling salesman tours" are white numbers. Red cell-colors correspond to instability.

Δc15															
1200	-629 (-629)	-529 (-529)	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	677 (770)
1100	-629 (-629)	-529 (-529)	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	677 (770)
1000	-629 (-629)	-529 (-529)	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	679 (770)
900	-629 (-629)	-529 (-529)	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	679 (770)
800	-629 (-629)	-529 (-529)	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	679 (770)
700	-629 (-629)	-529 (-529)	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	679 (770)
600	-629 (-629)	-529 (-529)	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	677 (770)
500	-629 (-629)	-529 (-529	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	677 (770)
400	-629 (-629)	-529 (-529	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	677 (770)
300	-629 (-629)	-529 (-529	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	679 (770)
200	-629 (-629)	-529 (-529	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	677 (770)
100	629 (-629)	-529 (-529)	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	670 (670)	677 (770)
0	-629 (-629)	-529 (-529)	-429 (-429)	-329 (-329)	-229 (-229)	-129 (-129)	-29 (-29)	70 (70)	170 (170)	270 (270)	370 (370)	470 (470)	570 (570)	610 (670)	610 (770)
-100	-667 (-629)	-503 (-529)	-467 (-429)	-367 (-329)	-267 (-229)	-103 (-129)	- 67 (-29)	32 (70)	132 (170)	232 (270)	332 (370)	332 (470)	510 (570)	510 (670)	510 (770)
-200	-767 (-629)	-667 (-529)	-567 (-429)	-467 (-329)	-367 (-229)	-267 (-129)	-167 (-29)	-67 (70)	32 (170)	132 (270)	232 (370)	332 (470)	410 (570)	410 (670)	410 (770)
	- 1200	- 1100	- 1000	-900	-800	-700	-600	-500	-400	-300	-200	-100	0	100	200
	Δc12														

in Figure 4a. The direct numerical simulation of (15) is now performed for different combinations of $(\Delta c_{12}, \Delta c_{15})$. The 3D stability analysis conducted here consists of solving (15) numerically for each combination of $(\Delta c_{12}, \Delta c_{15})$. TABLE 2 reports sample results of the 3D stability analysis. For each combination of perturbations Δc_{12} and Δc_{15}, the total costs of the "previously computed traveling salesman tours" (see black numbers in brackets in TABLE 2) and corresponding total costs of the "newly computed traveling salesman tours" (see white numbers) are calculated. The total costs are calculated according to (9) using the numerical solutions $\vec{x}$ of (15) for different combinations of $(\Delta c_{12}, \Delta c_{15})$. Full results of the 3D stability analysis are depicted in Figure 7. A specific solution $\vec{x}$ of (15) is obtained at coordinate $(\Delta c_{12}, \Delta c_{15}) = (+50, -50)$ (see circle surrounding a red star in Figure 7). At this coordinate the "newly computed TSP" is $x_9 \rightarrow x_{18} \rightarrow x_{11} \rightarrow x_{19} \rightarrow x_{25} \rightarrow x_8$ with total cost $c_{total} = 560.4$ $(\tilde{L} \approx 560.4)$.

We have also used (15) to obtain the "previously computed TSP tour" (or "target TSP tour") at coordinate $(\Delta c_{12}, \Delta c_{15}) = (+50, 0)$. The solution obtained is $x_9 \rightarrow x_{18} \rightarrow x_{11} \rightarrow x_{19} \rightarrow x_{25} \rightarrow x_8$ with total cost $c_{total} = 610.4$ $(\tilde{L} \approx 610.4)$. Comparing the total costs of the "previously computed TSP tour" with the total cost of the "newly computed TSP" it clearly appears that the "previously computed TSP tour" is unstable at coordinate $(\Delta c_{12}, \Delta c_{15}) = (+50, -50)$. Further we have extended the results of stability published in [11] by computing the area of instability of the previously computed TSP tour/solution.

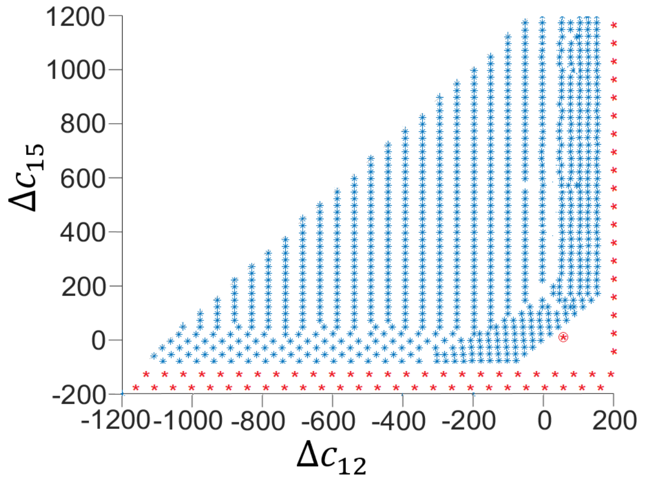


FIGURE 7. Results (obtained by the neuro-processor) of the stability analysis (in Fig. 3) in terms of Δc_{12} and Δc_{15}. This figure is obtained through the numerical solving of (15). Stability region (blue stars); a specific point of instability (see circle surrounding a red star); several other points of instability are found in the area surrounding the region of stability. These results complete those in Fig. 6 (say in [11]) by providing area of instability.

Sample points of instability are reported in TABLE 2 along with values of the total costs of their corresponding TSP tours/solutions. The points at which the "previously computed TSP tour" is unstable are reported in red cells of TABLE 2. These points are also depicted/represented with red stars in Figure 7. Overall, comparing the results published in [11] (see Figure 6) with results obtained using the neuro-processor solver model in (15) (see Figure 7) a very good agreement is obtained between them. This agreement validates and witnesses the effectiveness of the neuro-processor based TSP solver concept modeled by (15).

C. BIFURCATION ANALYSIS AND CONVERGENCE TO THE OPTIMAL TSP SOLUTION OF THE NEURO- PROCESSOR DEVELOPED

The aim of this section is to demonstrate that a suitable (or appropriate) choice of the internal parameters α, β_1, β_2, β_4, and $\beta_{5,I}$ of the neuro-processor based TSP solver modeled by system (15) can guarantee optimality of the TSP solution. The core objective here is to demonstrate that it is possible to configure internal processor parameters such that the neuro-processor remains stable and always-converging towards the exact TSP tour for any input graph network. Therefore, the concept of bifurcation analysis (with regards to the neuro-processor internal parameters α, β_1, β_2, β_4, and β_5) is introduced as a systematic and appropriate framework that can be used to perform a suitable choice of values and windows/ranges of the key internal parameters of (15) in order to guarantee the optimality of the obtained TSP solution. Hereby, after a stable values-range of internal processor parameters has been selected, various changes of the inputs graphs are considered (say, changes in arc weights and or graph dimensions (size, magnitude, topology)); thereby numerically/experimentally check if the convergence to the exact TSP solutions is always obtained.

The general applicability of the above mentioned bifurcation concept is practically illustratively demonstrated (or proven) through consideration of the following scenarios: (a) original graph network with unperturbed costs of edges; (b) graph network subject to random perturbation of costs of edges; (c) modification of graph topology (e.g. suppression of edges, addition of edges, etc..); (d) case of huge size graph network, etc. For all above scenarios, we use same (or identical) ranges/windows of neuro-processor internal parameters α, β_1 for investigation. These two parameters are used as bifurcation control parameters. The outcome of investigation reveals that, in the proposed windows/ranges of α and β_1, the neuro-processor developed always converges to the optimal TSP solution when considering each scenario. This witnesses the guaranty of sure convergence to the optimal TSP tour when performing in appropriate windows of α and β_1.

a. ***Scenario 1***, *reference scenario: Bifurcation analysis in a graph network with unperturbed costs of edges (constant costs of edges)*

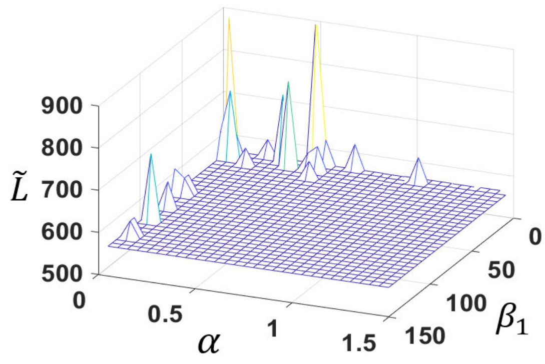


FIGURE 8. Results of the 2D bifurcation analysis in terms of α (step size of decision neurons) and β_1 (step size of multiplier neurons). TSP solutions with different total costs $\tilde{L}$ are shown. (a) Flat area with blue color: This is the domain of sure convergence to the global minimum ($\tilde{L} = 570.3$). This corresponds to the optimal TSP solution. (b) Area with pulses: This is the domain of convergence to local minima. Several local minima are detected ($\tilde{L} = 610.4$; $\tilde{L} = 629.4$; $\tilde{L} = 633.4$; $\tilde{L} = 731.6$; $\tilde{L} = 774.5$; $\tilde{L} = 884.9$). This plot witnesses the necessity of controlling optimality of TSP solution in order to avoid local minima and ensure convergence to the global minimum.

The bifurcation analysis is performed using constant values of costs of edges $U = \left[c_1, c_2, \ldots, c_{29}, c_{30}\right]^T$ as defined in the text (see case of Figure 3). Coefficients α, β_1 of (15) are used as control parameters (also called bifurcation parameters). Following parameters with constant values are also used for bifurcation analysis: $\beta_2 = 100$, $\beta_3 = 100, \beta_3^* = 100$, $\beta_4 = 10$, and $\beta_5 = 0.1$. Step-size used for numerical solving of (15) is $h = 0.0025$. Remaining parameters A, D, and S of (15) are deduced from graph topology in Figure 3 according to general expression (15h). The bifurcation analysis is performed numerically by monitoring control parameters in ranges $0 < \alpha \leq 1.5$ and $0 < \beta_1 \leq 150$. Results obtained as numerical solution of (15) are depicted in Figure 8. As it appears in Figure 8, a TSP solution with smallest total cost $\tilde{L} = 570.3$ is obtained in a wide/large region/domain of variation of α and β_1 (see region with blue color in Figure 8). The TSP solution obtained (with total cost $\tilde{L} = 570.3$) corresponds to the optimal TSP tour. Figure 8 also reveals that optimality of the TSP solution is not obtained (or preserved) for some specific values of control parameters α, β_1. This is clearly illustrated by pulses depicted in Figure 8. These pulses express several other (alternative) TSP solutions with total costs $\tilde{L} = 610.4$, $\tilde{L} = 629.4$, $\tilde{L} = 633.4$, $\tilde{L} = 731.6$, $\tilde{L} = 776.3$, and $\tilde{L} = 899.4$, etc. This witnesses the fact that a suitable (or appropriate) choice of key parameters of (15) is of necessary importance to guarantee optimality of the TSP solution.

The issue related to a sure convergence to the optimal TSP tour/solution is addressed in next case studies (in Sections (b), (c) and (d)) through various numerical simulations.

An important remark can be expressed concerning results of the bifurcation analysis in Figure 8. Indeed, it appears from Figure 8 that a suitable choice of values of parameters for the numerical solving of (15) can guarantee/ensure optimality of

TABLE 3. Values of perturbations applied to four arcs weights in Figure 3 and results of the optimal TSP tours obtained for each set as numerical solutions of the neuro-processor model in (15).

	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6	Set 7
Δc_{12}	+100	0	-100	-200	-300	-400	-500
Δc_{21}	+100	0	-100	-200	-300	-400	-500
Δc_{15}	+100	0	+100	+200	+300	+400	+500
Δc_{51}	+100	0	+100	+200	+300	+400	+500
$\tilde{L}_i$	670.3	570.3	470.3	370.3	270.3	170.3	70.3

TABLE 4. Perturbation of height arcs weights in Figure 3 and results of the optimal TSP tours obtained/detected as numerical solutions of the neuro-processor model in (15).

	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6	Set 7
Δc_{12}	+100	0	-100	-200	-300	-400	-500
Δc_{21}	+100	0	-100	-200	-300	-400	-500
Δc_{15}	+100	0	+100	+200	+300	+400	+500
Δc_{51}	+100	0	+100	+200	+300	+400	+500
Δc_{46}	+100	0	-100	-200	-300	-400	-500
Δc_{64}	+100	0	-100	-200	-300	-400	-500
Δc_{14}	+100	0	+100	+200	+300	+400	+500
Δc_{41}	+100	0	+100	+200	+300	+400	+500
$\tilde{L}_i$	670.3	570.3	433.4	233.4	33.4	-166,6	-366.6

TABLE 5. Perturbation of twelve arcs weights in Figure 3 and results of the optimal TSP tours obtained/detected as numerical solutions of the neuro-processor model in (15).

	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6	Set 7
Δc_{12}	+100	0	-100	-200	-300	-400	-500
Δc_{21}	+100	0	-100	-200	-300	-400	-500
Δc_{15}	+100	0	+100	+200	+300	+400	+500
Δc_{51}	+100	0	+100	+200	+300	+400	+500
Δc_{46}	+100	0	-100	-200	-300	-400	-500
Δc_{64}	+100	0	-100	-200	-300	-400	-500
Δc_{14}	+100	0	+100	+200	+300	+400	+500
Δc_{41}	+100	0	+100	+200	+300	+400	+500
Δc_{24}	+100	0	-100	-200	-300	-400	-500
Δc_{42}	+100	0	-100	-200	-300	-400	-500
Δc_{25}	+100	0	+100	+200	+300	+400	+500
Δc_{52}	+100	0	+100	+200	+300	+400	+500
$\tilde{L}_i$	670.3	570.3	433.4	233.4	-53.1	-353.1	-653.1

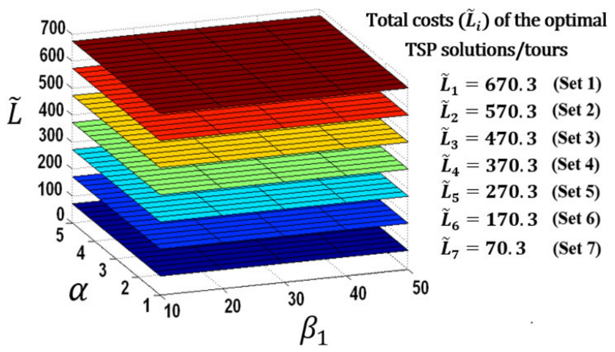


FIGURE 9. Results of the 2D bifurcation analysis in terms of α (step size of decision neurons) and β_1 (step size of multiplier neurons). Seven sets of perturbations (Δc_{12}, Δc_{21}, Δc_{15}, Δc_{51}) are used as reported in TABLE 3. For each set, it clearly appears that the total cost ($\tilde{L}_i$) of the TSP tour detected remains constant for all values of α and β_1 selected in windows $1 \le \alpha \le 5$ and $10 < \beta_1 \le 50$. Further, it has been found that each value $\tilde{L}_i$ corresponds to the global minimum as each $\tilde{L}_i$ corresponds to the smallest cost. This guarantees optimality of the TSP solution for $1 \le \alpha \le 5$ and $10 < \beta_1 \le 50$.

the TSP solution/tour. Therefore, the main objective of the analysis carried out in the next sections (see below) is to demonstrate numerically that a suitable choice of parameter-values in (15) can guarantee optimality of the TSP solution/tour for any input graph, independently of its related parameters (i.e. size, magnitude, topology, and arc weights values). We demonstrate through various numerical simulations the existence of neuro-processor solver parameter values-ranges (windows) under which the optimality of the TSP solution is guaranteed (ensured). A related stress-testing is performed through examining various scenarios envisaged as case-studies. The first scenario considers a graph network with several edges subject to random perturbations (see TABLES 3, 4 and 5). The second scenario is concerned with the variation of the graph topology. In all these scenarios envisaged it is demonstrated, while using the same (or identical) value-ranges of internal neuro-processor parameters of (15), that the neuro-processor solver always detect the optimal TSP tour. The results confirming the optimality of the TSP tour detected are depicted in Figures. 8, 9, 10, 11, 15, 19, and 22.

b. *Scenario 2: Bifurcation analysis for a graph network subject to a random perturbation of costs of edges and optimality of the TSP solution/tour*

The bifurcation analysis is now performed for the case of a graph network subject to perturbations of selected arc weights. The coefficients α, β_1 of (15) are used as control parameters defined in the respective ranges $1 \le \alpha \le 5$ and $10 \le \beta_1 \le 50$. The following values of parameters are used for numerical simulations of (15): $\beta_2 = 100, \beta_3 = 150, \beta_3^* = 50$, $\beta_4 = 10$, $\beta_5 = 0.045$, and $h = 0.00025$. The bifurcation analysis considers three different cases; *case 1*- perturbation of four edges of the graph (see TABLE 3); *case 2*- perturbation of eight edges of the graph (see TABLE 4); *case 3*- perturbation of twelve edges (see TABLE 5). **Considering case 1.** Four edges of Figure 3 are perturbed as follows: $c_{1\to 2}$ $(c_1 + \Delta c_{12})$, $c_{2\to 1}$ $(c_2 + \Delta c_{21})$, $c_{1\to 5}$ $(c_9 + \Delta c_{15})$, $c_{5\to 1}$ $(c_{10} + \Delta c_{51})$. We use seven different strengths of perturbations denoted by $Set_i = (\Delta c_{12}, \Delta c_{21}, \Delta c_{15}, \Delta c_{51})$ $(i = 1, \ldots, 7)$. The values of all Set_i are reported in TABLE 3. The numerical solving of (15) is performed and results are depicted in Figure 9. It clearly appears from Figure 9 that each

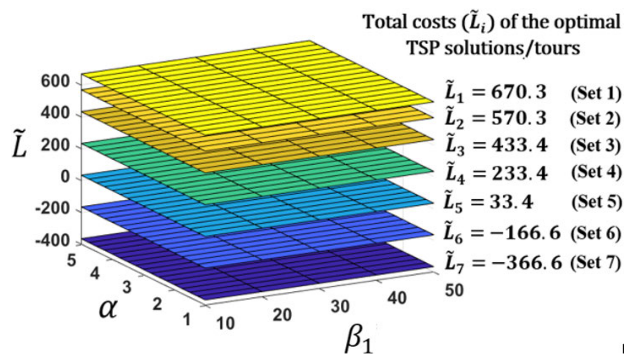


FIGURE 10. Results of the 2D bifurcation analysis in terms of α (step size of decision neurons) and β_1 (step size of multiplier neurons). Seven sets of perturbations (Δc_{12}, Δc_{21}, Δc_{15}, Δc_{51}, Δc_{46}, Δc_{64}, Δc_{14}, Δc_{41}) are used as reported in TABLE 4. For each set, it clearly appears that the total cost ($\tilde{L}_i$) of the TSP tour detected remains constant for all values of α and β_1 selected in windows $1 \le \alpha \le 5$ and $10 < \beta_1 \le 50$. Further, it has been found that each value $\tilde{L}_i$ corresponds to the global minimum as each $\tilde{L}_i$ corresponds to the smallest cost. This guarantees optimality of the TSP solution for $1 \le \alpha \le 5$ and $10 < \beta_1 \le 50$.

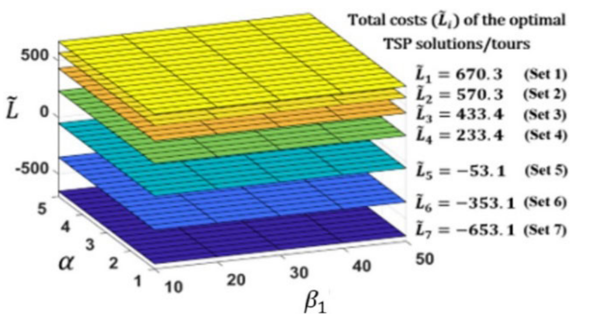


FIGURE 11. Results of the 2D bifurcation analysis in terms of α (step size of decision neurons) and β_1 (step size of multiplier neurons). Seven sets of perturbations (Δc_{12}, Δc_{21}, Δc_{15}, Δc_{51}, Δc_{46}, Δc_{64}, Δc_{14}, Δc_{41}, Δc_{24}, Δc_{42}, Δc_{25}, Δc_{52}) are used as reported in TABLE 5. For each set, it clearly appears that the total cost ($\tilde{L}_i$) of the TSP tour detected remains constant for all values of α and β_1 selected in windows $1 \le \alpha \le 5$ and $10 < \beta_1 \le 50$. Further, it has been found that each value $\tilde{L}_i$ corresponds to the global minimum as each $\tilde{L}_i$ corresponds to the smallest cost. This guarantees optimality of the TSP solution for $1 \le \alpha \le 5$ and $10 < \beta_1 \le 50$.

set of perturbation (Set_i) leads to a constant value of the total cost $\tilde{L}$ of the TSP tour detected. It also clearly appears from Figure 9 that for each set of perturbation Set_i (in TABLE 3) the value of $\tilde{L}$ remains constant and optimal for all parameter ranges $1 \le \alpha \le 5$ and $10 \le \beta_1 \le 50$. The constant value of $\tilde{L}$ obtained for each set corresponds to the smallest TSP cost. This justifies the convergence of the neuro-processor developed to the respectively optimal TSP solution.

Considering case 2. Eight edges of Figure 3 are perturbed as follows: $c_{1\to2}$ ($c_1 + \Delta c_{12}$), $c_{2\to1}$ ($c_2 + \Delta c_{21}$), $c_{1\to5}$ ($c_9 + \Delta c_{15}$), $c_{5\to1}$ ($c_{10} + \Delta c_{51}$), $c_{4\to6}$ ($c_{25} + \Delta c_{46}$), $c_{6\to4}$ ($c_{26} + \Delta c_{64}$), $c_{1\to4}$ ($c_7 + \Delta c_{14}$), and $c_{4\to1}$ ($c_8 + \Delta c_{41}$). We use seven different strengths of perturbations denoted by $Set_i = (\Delta c_{12}, \Delta c_{21}, \Delta c_{15}, \Delta c_{51}, \Delta c_{46}, \Delta c_{64}, \Delta c_{14}, \Delta c_{41})$ $(i = 1, 2, \ldots, 7)$. The values of all Set_i are reported in TABLE 4. The numerical solving of (15) is performed and results are depicted in Figure 10. It clearly appears from Figure 10 that each set of perturbations (Set_i) leads to a constant value of the total cost $\tilde{L}$ of the TSP tour detected. It also clearly appears from Figure 10 that for each set of perturbations Set_i (in TABLE 4) the value of $\tilde{L}$ remains constant for all values of α and β_1 selected in windows $1 \le \alpha \le 5$ and $10 \le \beta_1 \le 50$. The constant value of $\tilde{L}$ obtained for each set corresponds to the smallest TSP cost. This justifies the convergence of the neuro-processor developed to the optimal TSP solution.

Considering case 3. Twelve arc weights of Figure 3 are now perturbed as follows: edges $c_{1\to2}$ ($c_1 + \Delta c_{12}$), $c_{2\to1}$ ($c_2 + \Delta c_{21}$), $c_{1\to5}$ ($c_9 + \Delta c_{15}$), $c_{5\to1}$ ($c_{10} + \Delta c_{51}$), $c_{4\to6}$ ($c_{25} + \Delta c_{46}$), $c_{6\to4}$ ($c_{26} + \Delta c_{64}$), $c_{1\to4}$ ($c_7 + \Delta c_{14}$), $c_{4\to1}$ ($c_8 + \Delta c_{41}$), $c_{2\to4}$ ($c_{15} + \Delta c_{24}$), $c_{4\to2}$ ($c_{16} + \Delta c_{42}$), $c_{2\to5}$ ($c_{13} + \Delta c_{25}$), and $c_{5\to2}$ ($c_{14} + \Delta c_{52}$). We use seven different strengths of perturbations denoted by $Set_i = (\Delta c_{12}, \Delta c_{21}, \Delta c_{15}, \Delta c_{51}, \Delta c_{46}, \Delta c_{64}, \Delta c_{14}, \Delta c_{41}, \Delta c_{24}, \Delta c_{42}, \Delta c_{25}, \Delta c_{52})$.

The values of all Set_i are reported in TABLE 5. The numerical solving of (15) is performed and results are depicted in Figure 11. It clearly appears from Figure 11 that each set of perturbation (Set_i) leads to a constant value of the total cost $\tilde{L}$ of the TSP tour detected. It also clearly appears from Figure 11 that for each set of perturbation Set_i (in TABLE 5) the value of $\tilde{L}$ remains constant for all $1 \le \alpha \le 5$ and $10 \le \beta_1 \le 50$. The constant value of $\tilde{L}$ obtained for each set corresponds to the smallest TSP cost. This justifies convergence of the neuro-processor developed to the optimal TSP solution.

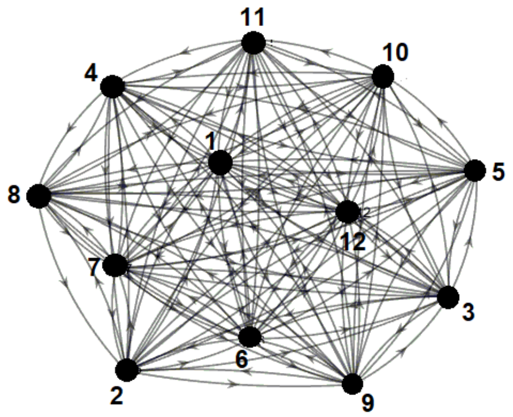
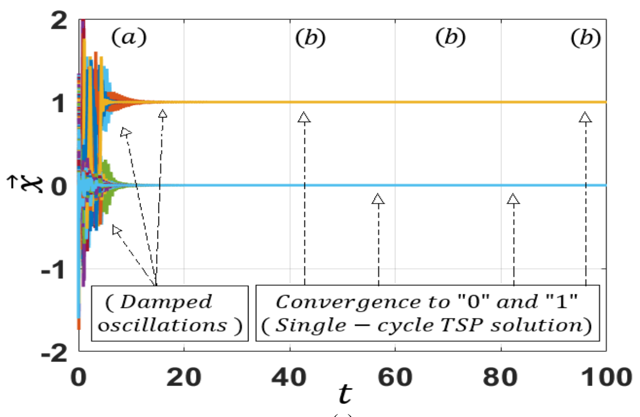


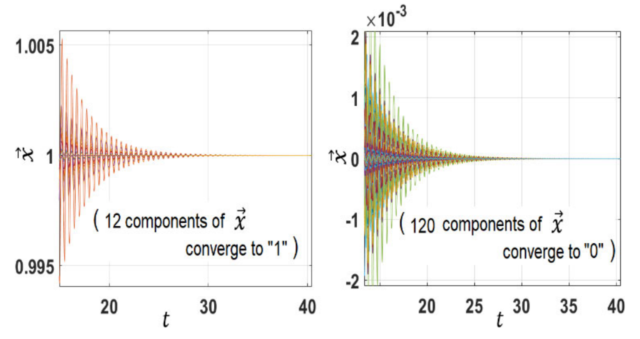
FIGURE 12. A complete graph network of magnitude "12" and size "132". Edges are bi-directional and their costs are obtained through the expression $c_i = i$ ($i = 1, 2, \ldots, 132$ stand for the indexes of edges). The optimal TSP tour is obtained as numerical solution of (15). The values of parameters used to solve (15) are defined as follows: $\beta_2 = 100$, $\beta_3 = 150$, $\beta_3^* = 50$, $\beta_4 = 10$, $\beta_5 = 0.045$, $h = 0.00025$ (step size) $1 \le \alpha \le 5$ and $10 < \beta_1 \le 50$.

The study carried out in this sub-section was related to a numerical study of bifurcation analysis in presence of "several edges of the graph subject to weights perturbations". This study has shown/confirmed that all-over the fixed windows of control parameters ($1 \le \alpha \le 5$ and $10 \le \beta_1 \le 50$) optimality of the TSP solution is always reached/ensured.

This sub-section has demonstrated that optimality of the TSP solution (obtained using the neuro-processor model in (15)) is preserved even in case of graph networks subject to



(a)



(b)

FIGURE 13. **(a) Numerical solution $\vec{x}$ of (15). The components of $\vec{x}$ are binary values which are used to detect the TSP solution/tour in Figure 12. (*a*) Damped oscillations in range $0 < t < 35$. (*b*) Permanent phase corresponding to the final TSP solution in form of a single-cycle obtained for all $t > 35$. This latter solution $\vec{x}$ of (15) is stable as it remains unchanged for all values of t (say $t > 35$). The optimality of the TSP solution/tour obtained here is confirmed by the 2D bifurcation diagram in Figure 15. (b) A zooming of the solution $\vec{x}$ in Figure 13a in the range $10 < t < 40$. This range overlaps the range of damped oscillations ($0 < t < 35$) and the range of permanent dynamics ($t > 35$). The zooming clearly illustrates the convergence of $\vec{x}$ to binary variables "0" and "1". When t increases further ($t > 35$) all solutions $\vec{x}$ merge into two lines: Line 1: $x_i = 1$ (i = 14, 17, 38, 39, 51, 60, 69, 74, 79, 84, 92, and 101) and Line 2: All others 120 components are equal zero ($x_i = 0$). This witnesses the binarization of solutions $\vec{x}$ of (15).**

random perturbation of costs of edges for fixed value-ranges of control parameters (say, $1 \leq \alpha \leq 5$ and $10 \leq \beta_1 \leq 50$), which have been obtained/selected through various numerical simulations from the reference scenario (see Figure 8). Several sets of perturbations applied to costs of edges have been considered and it has been demonstrated that optimality of the TSP solution is always ensured in the defined/fixed ranges for α and β_1.

The next sub-section investigates the optimality of TSP solutions in presence of graph networks of different topologies. The main objective of our investigation in this sub-section is to demonstrate that for a selected parameters-values window/range for α and β_1 (selected from Figure 8) the optimality of the TSP solution regardless of graph size, graph magnitude and graph topology is still maintained. The investigation is focused on a bifurcation analysis with regards to α and β_1. The analysis considers various graphs with different topologies. The aim is to show

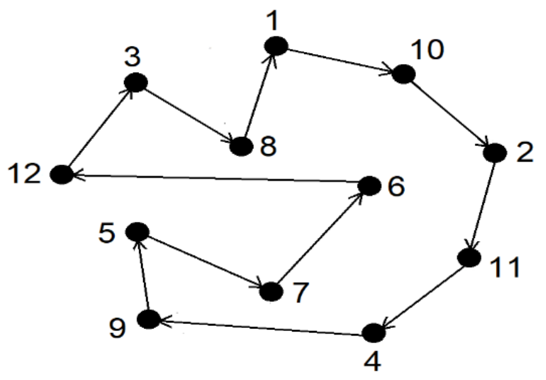


FIGURE 14. **TSP tour obtained according to the numerical solution $\vec{x}$ for the graph in Figure 12. The TSP tour detected corresponds to $x_i = 1$ (i=14, 17, 38, 39, 51, 60, 69, 74, 79, 84, 92, and 101)). The cost of an edge is taken (in the example) equal to the edge index. Thus, using x_i in (9) leads to the following total cost of the TSP tour $\tilde{L} = \sum (i * x_i) = 718$.**

that all-over that fixed windows $1 \leq \alpha \leq 5$ and $10 \leq \beta_1 \leq 50$ the optimality of TSP is always reached for all those multiple graph-topologies. As illustrative examples, two scenario-settings are thereby investigated. The first setting considers a graph of magnitude "12" and size "132". The second setting does consider a graph of magnitude "24" and size "552".

c. *Scenario 3: Bifurcation analysis in graph networks subject to changes in their topologies and optimality of the TSP solution/tour*

The bifurcation analysis is now performed in graphs with different topologies. Two scenarios are considered, namely: scenario-setting-1- a graph network of magnitude "12" and size "132", and scenario-setting-2- a graph of magnitude "24" and size "552". For the two scenarios, our investigations use exactly the same values and same windows/ranges of the control parameters as in the previous section (a neuro-processor parameter-values range/window which is selected from the reference scenario in/of Figure 8, where it is seen that for that range/window the solution is always stable and optimal).

Scenario setting 1. We now consider the topology of a complete graph of magnitude "12" and size "132" (see Figure 12). All nodes in Figure 12 are pair-wisely connected by/through bi-directional edges. In Figure 12 the costs of edges are obtained from the expression $c_i = i$, where i stands for the edge- index.

The TSP solution in Figure 13a is obtained as numerical solution $\vec{x}$ of (15) for the graph of Figure 12. We use same/identical values and/or windows of the control parameters as in Figures 9, 10 and 11. The choice of the control parameters $\alpha = 1$ and $\beta_1 = 10$ has led to results depicted in Figures 13a and 13b. The zooming in Figure 13b shows the convergence of "12" components of $\vec{x}$ to "1" and also convergence of remaining "120" components of $\vec{x}$ to "0". This witnesses the binarization of solutions $\vec{x}$ in (15). The non-zero components of $\vec{x}$ are further used to obtain the TSP tour depicted in Figure 14. The optimality of the TSP tour obtained

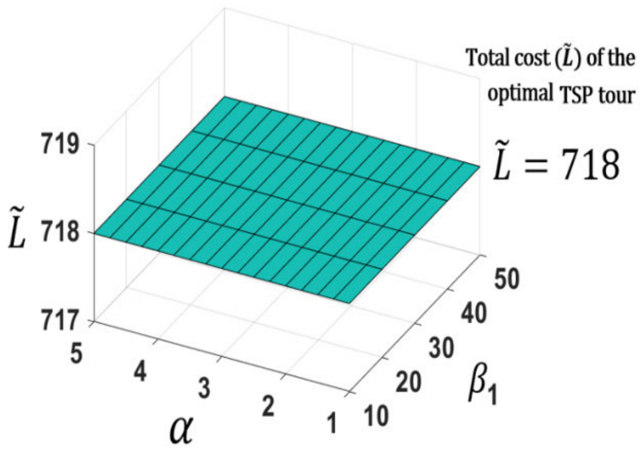


FIGURE 15. Results of the 2D bifurcation analysis in terms of α and β_1 in a graph of magnitude 12 and size 132 (see Figure 12). The total cost ($\tilde{L}$) of the TSP tour detected remains constant for all values of α and β_1 selected in windows $1 \leq \alpha \leq 5$ and $10 < \beta_1 \leq 50$. Further, it has been found through our various numerical simulations that $\tilde{L}$ corresponds to the global minimum. Thus, an optimality of the TSP tour is ensured in selected windows of α and β_1.

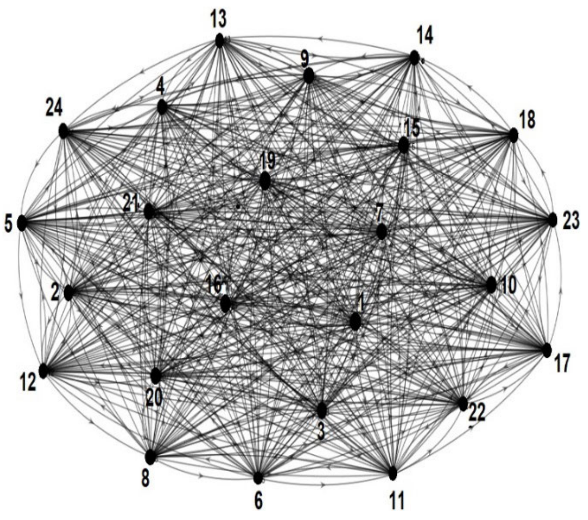


FIGURE 16. A complete graph network of magnitude "24" and size "552". Edges are bi-directional and their costs are obtained through the expression $c_i = i$ ($i = 1, 2, \ldots, 552$ stand for the indexes of edges). The optimal TSP tour is obtained as numerical solution of (15). The values of parameters used to solve (15) are defined as follows: $\beta_2 = 100$, $\beta_3 = 150, \beta_3^* = 50$, $\beta_4 = 10, \beta_5 = 0.045$, $h = 0.00025$ (step size) $1 \leq \alpha \leq 5$ and $10 < \beta_1 \leq 50$.

is further addressed through a thoroughly bifurcation analysis. This analysis uses exactly same settings/windows of the control parameters as in Figures 9, 10 and 11. The outcome (results) of the bifurcation analysis is depicted in Figure 15. As it appears from Figure 15 the unicity of the TSP tour is preserved in the domains of variation of control parameters ($1 \leq \alpha \leq 5$ and $10 < \beta_1 \leq 50$). This confirms the optimality of the TSP solution/tour detected using (15).

Scenario setting 2. We now consider the topology of a complete graph of magnitude "24" and size "552" (see Figure 16). All nodes in Figure 16 are pair-wisely

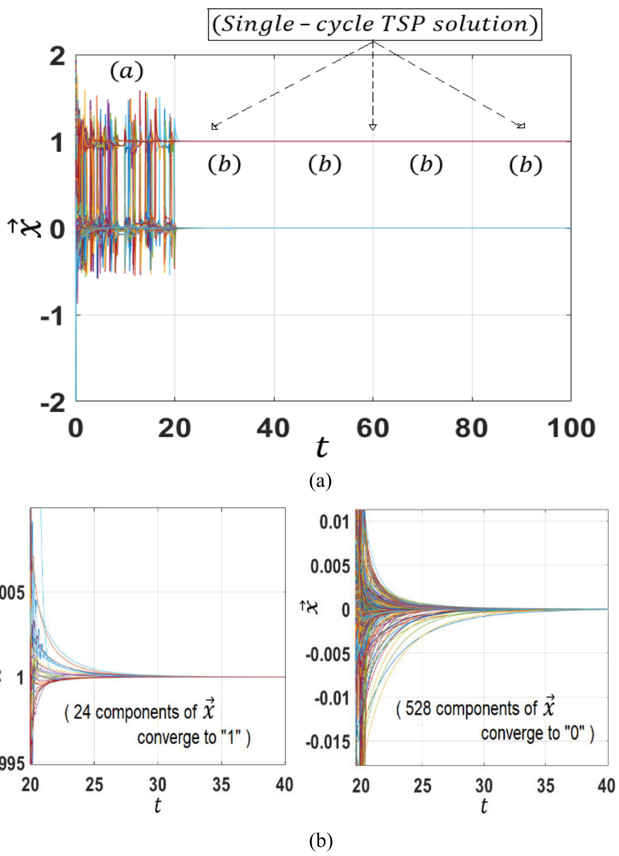


FIGURE 17. (a) Numerical solution $\vec{x}$ of (15). The components of $\vec{x}$ are binary values which are used to detect the TSP solution/tour in Figure 16. (*a*) Transient phase characterized by oscillatory dynamics in range $0 < t < 20$. (*b*) Permanent phase corresponding to the final TSP solution in form of a single-cycle obtained for all $t > 20$. This latter solution $\vec{x}$ of (15) is stable as it remains unchanged for all values of t (say $t > 20$). The optimality of the TSP solution/tour obtained here is confirmed by Figure 19. (b) A zooming of the solution $\vec{x}$ in Figure 17a in the range $20 < t < 40$. This range overlaps the range of transient dynamics ($20 < t < 35$) and the range of permanent dynamics ($t > 35$). The zooming clearly illustrates the convergence of $\vec{x}$ to binary variables "0" and "1". When t increases further ($t > 35$) all solutions $\vec{x}$ merge into two lines: Line 1: $x_i = 1$ (i=27, 32, 84, 85, 114, 115, 151, 160, 195, 200, 230, 243, 257, 276, 303, 312, 329, 340, 350, 369, 374, 385, 412 and 413) and Line 2: All others 528 components are equal zero ($x_i = 0$). This witnesses the binarization of solutions $\vec{x}$ of (15).

connected by/through bi-directional edges. In Figure 16 the costs of edges are obtained from the expression $c_i = i$, where i stands for the edge- index.

The TSP solution for the graph in Figure 16 is obtained as numerical solution $\vec{x}$ of (15). We use same/identical values and/or windows/ranges of the control parameters as in Figures 9, 10, 11 and 15. The choice of control parameters $\alpha = 1$ and $\beta_1 = 10$ has led to results depicted in Figures 17a and 17b. The zooming in Figure 17b shows the convergence of "24" components of $\vec{x}$ to "1" and also convergence of remaining "528" components of $\vec{x}$ to "0". This witnesses the binarization of solutions $\vec{x}$ in (15). The non-zero components of $\vec{x}$ are further used to obtain the TSP tour depicted in Figure 18. The optimality of the TSP tour obtained is further addressed through a thoroughly bifurcation analysis.

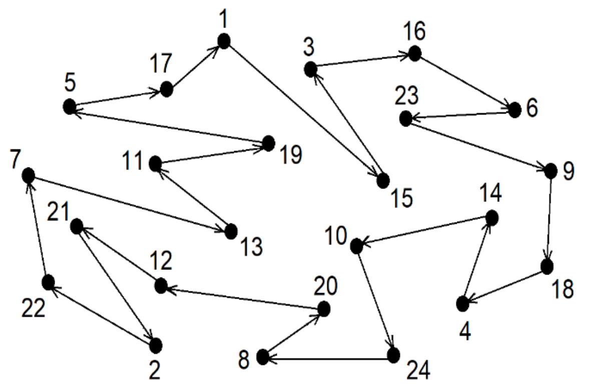


FIGURE 18. TSP tour obtained according to the numerical solution $\vec{x}$ for the graph in Figure 16. The TSP tour detected corresponds to $x_i = 1$ (i=27, 32, 84, 85, 114, 115, 151, 160, 195, 200, 230, 243, 257, 276, 303, 312, 329, 340, 350, 369, 374, 385, 412 and 413). The cost of an edge is equal to the edge index. Thus using values of x_i in (9) leads to the total cost of TSP tour $\tilde{L} = \sum (i * x_i) = 5756$.

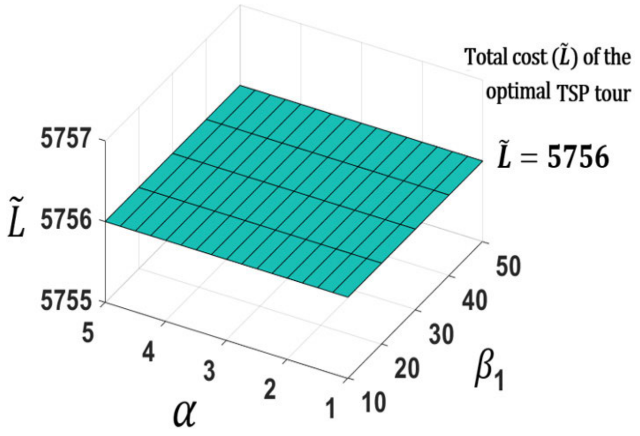


FIGURE 19. Results of the 2D bifurcation analysis in terms of α and β_1 in a graph of magnitude 24 and size 552 (see Figure 16). The total cost ($\tilde{L}$) of the TSP tour detected remains constant for all values of α and β_1 selected in windows $1 \leq \alpha \leq 5$ and $10 < \beta_1 \leq 50$. Further, it has been found through our various numerical simulations that $\tilde{L}$ corresponds to the global minimum. Thus, an optimality of the TSP tour is ensured in selected windows of α and β_1.

This analysis uses exactly same settings/windows of the control parameters as in Figures 9, 10, 11, and 15. The outcome (results) of the bifurcation analysis is depicted in Figure 19. As it appears from Figure 19 the unicity of the TSP tour is preserved in the domains of variation of control parameters ($1 \leq \alpha \leq 5$ and $10 < \beta_1 \leq 50$). This confirms the optimality of the TSP solution/tour detected using the neuro-processor model (15).

D. APPLICATION EXAMPLE 2 (PUBLISHED IN [26]): COMPUTATION OF THE TSP TOUR, BENCHMARKING WITH THE NEURO- PROCESSOR DEVELOPED, AND OPTIMALITY OF THE TSP SOLUTION/TOUR

The objectives of the study in this section are twofold. The first concern is to demonstrate that the neuro-processor concept developed in (15) can be used to detect the TSP tour

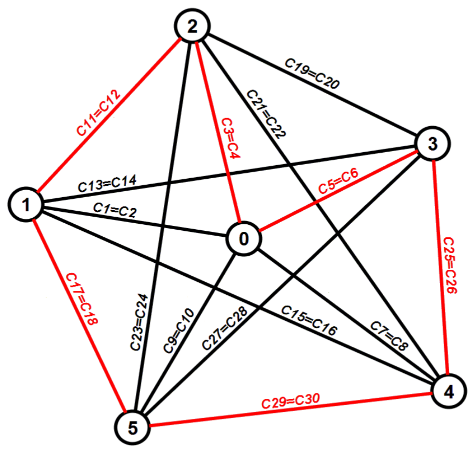


FIGURE 20. A city graph with distance weights (this case study is published in [26]). The costs of edges c_i are defined in the text. The corresponding total cost of the "traveling salesman tour" (see red lines) is equal to $c_{total} = 159$.

obtained in [26] and materialized by the cyclic trajectory in red color in Figure 20. The second concern is to demonstrate that the optimality of TSP solution is ensured (guaranteed) by the neuro-processor. Thus a bifurcation analysis is carried out to demonstrate that the parameter-values-ranges and windows/ranges of parameters used in the previous examples to guarantee optimality of the TSP tour can also be used in case of the graph network in [26] (see also Figure 20). Various numerical simulations are considered (as stress-testing) to confirm the optimality of the TSP solution/tour under defined/specific values and ranges/windows of system parameters.

In order to facilitate the benchmarking between the method in [26] and the neuro-processor concept developed in this paper (see model (15)) we use exactly identical values assigned to the costs of edges in [26]. The weights values in [26] are chosen as follows: $c_1 = c_2 = 25$; $c_3 = c_4 = 25$; $c_5 = c_6 = 22$; $c_7 = c_8 = 25$; $c_9 = c_{10} = 28$; $c_{11} = c_{12} = 25$; $c_{13} = c_{14} = 45$; $c_{15} = c_{16} = 45$; $c_{17} = c_{18} = 35$; $c_{19} = c_{20} = 30$; $c_{21} = c_{22} = 55$; $c_{23} = c_{24} = 50$; $c_{25} = c_{26} = 25$; $c_{27} = c_{28} = 50$; $c_{29} = c_{30} = 27$. Furthers, the values of parameters used to solve (15) numerically are identical to the values used in Figures 9, 10, 11, 15 and 19.

Using the aforementioned values of system parameters the numerical solving of (15) has led to results depicted in Figure 21. The results depicted in Figure 21 correspond to the TSP tour $x_3 \rightarrow x_5 \rightarrow x_{25} \rightarrow x_{29} \rightarrow x_{17} \rightarrow x_{11}$. This tour corresponds to the optimal "traveling salesman tour" with the total cost equal to $c_{total} = 159$. This value corresponds to the global minimum. Thus, the tour represents the "exact traveling salesman tour" under the predefined fundamental parameters of the city graph in Figure 20.

The benchmarking performed in TABLE 6 shows that the neuro-processor based TSP solver concept developed leads to same results as those published in [26]. Under the same parameter settings used by the method published in [26]

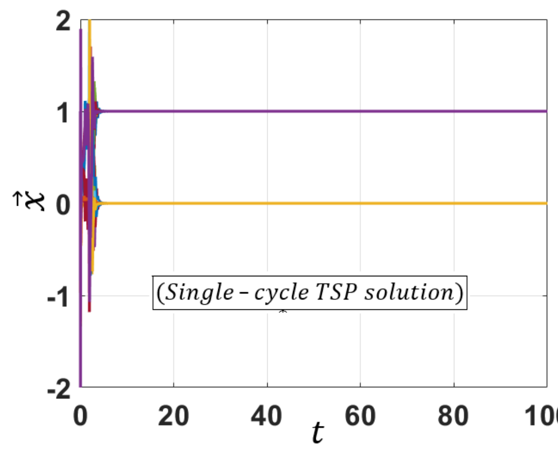


FIGURE 21. Numerical solution $\vec{x}$ of (15). The components of $\vec{x}$ are binary values which are used to detect the TSP solution/tour in Figure 20. A short transient phase characterized by damped oscillations is observed in the range $0 < t < 5$. When $t > 5$ a permanent phase corresponding to the final TSP solution in form of a single-cycle is obtained (for all $t > 5$). This latter solution $\vec{x}$ of (15) is stable as it remains unchanged for all values of t (say, $t > 5$). The optimality of the TSP solution/tour obtained here is confirmed by Figure 22.

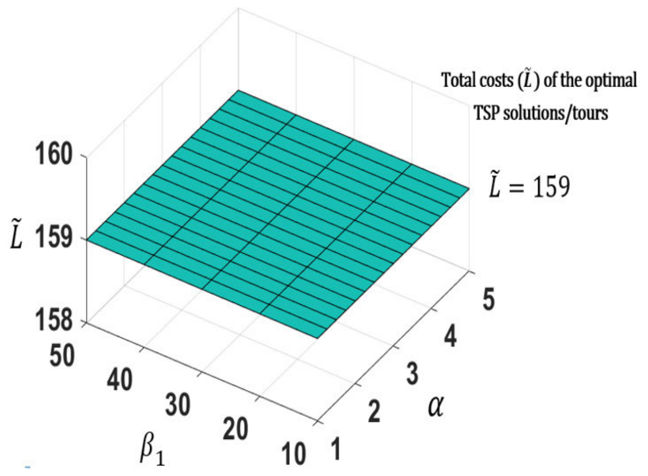


FIGURE 22. Results of the 2D bifurcation analysis in terms of α and β_1 in the graph of Figure 20. The total cost ($\tilde{L}$) of the TSP tour detected remains constant for all values of α and β_1 selected in windows $1 \leq \alpha \leq 5$ and $10 < \beta_1 \leq 50$. Further, it has been found through our various numerical simulations that $\tilde{L}$ corresponds to the global minimum. Thus, an optimality of the TSP tour is ensured in selected windows of α and β_1.

and the neuro-processor based concept, the same TSP solution/tour has been obtained.

The convergence of both methods is also reported in TABLE 6. However, concerning the iterations to achieve convergence, the neuro-processor has experienced (did need) about t = 5 iterations (see also in Figure 21). The time to convergence is not reported in [26] as mentioned in TABLE 6.

We now carry out the bifurcation analysis using identical (same) values of parameters as in Figure 21. The bifurcation diagram obtained in terms of α and β_1 as result of the numerical solution $\vec{x}$ of (15) is shown in Figure 22. This figure confirms the optimality of the TSP solution in the selected windows of control parameters α and β_1.

TABLE 6. Benchmarking results for application example 2: computation of the traveling salesman tour provided by results in Figure 21. This traveling salesman tour is represented with red lines in Figure 20.

Concepts	Exact TSP solution	Convergence	Convergence starts at t
Concept in [26]	$x_3 \rightarrow x_5 \rightarrow x_{25} \rightarrow x_{29} \rightarrow x_{17} \rightarrow x_{11}$	**Yes** (The same optimal TSP solution)	**Not reported in [26]**
neuro-processor concept developed	$x_3 \rightarrow x_5 \rightarrow x_{25} \rightarrow x_{29} \rightarrow x_{17} \rightarrow x_{11}$	**Yes** (The same optimal TSP solution)	$t \approx 5$ (see Fig. 21)

VI. CONCLUSION

This work has developed and validated a neuro-processor based dynamically reconfigurable TSP solver concept for the efficient and robust solving of single-cycle TSP in dynamically reconfigurable network graphs. The developed concept has been demonstrated as being capable of efficiently handling single-cycle TSP in undirected graphs and in bidirectional graphs, even in the case of large values (i.e. various settings – from very small to very large) of the edge costs. Furthers, it has been demonstrated that the developed concept can efficiently solve single-cycle TSP even in particular cases where several TSP paths with identical minimum total cost are identified within the graph. This novel concept does significantly contribute to the enrichment of the relevant state-of-the-art regarding TSP solving, since most traditional TSP solver concepts and algorithms cannot efficiently tackle such last-named situations and nor always ensure thereby a sure convergence. The efficiency in those various hard contexts is related to the core performance criteria at stake, namely stability, robustness, and fast convergence towards the optimal TSP solution.

One further most important contribution of this paper is that we have carried out the (local) stability analysis of equilibrium points and analytical conditions have been obtained under which the equilibrium points are stable. The analytical conditions derived have been further used to guarantee the sure "always"-convergence of the neuro-processor based TSP solver concept to the exact/true single-cycle TSP solution. This sure convergence pending the fulfillment of the analytical stability conditions established has been demonstrated/validated through extensive bifurcation analysis related illustrative examples.

Overall, the accuracy and the robustness/stability of the neuro-processor developed have been proven both analytically and numerically.

In the validation process, an extensive benchmarking has been carried out. Part of the results of this paper have been compared with the related results presented in [11] and [26]. The outcomes of this comparison have confirmed the effectiveness and correctness of the neuro-processor based single-cycle TSP solver concept developed here to efficiently tackle traveling salesman problems.

A series of ongoing works under consideration do relate amongst others to the following issues: (a) The experimental validation of the neuro-processor developed in this paper for the efficient solving of single-cycle TSP in large-scale and dynamically reconfigurable graph networks. The main concern here is to use the neuro-processor to solve a series of practical and real-world examples or real-life instances (e.g., standard TSP instances from TSPLIB). The validation will be carried out in the frame of a comprehensive benchmarking procedure consisting of comparing the results obtained by the neuro-processor with the results obtained for TSPLIB instances by involving some traditional and commonly used TSP algorithms/heuristics; (b) The extension of the concept developed (in this paper) to the case of non-additive (i.e. non-linear) path cost (i.e. the so-called non-additive shortest path (NASP) problem which asks for finding an optimal path that minimizes a certain multi-attribute non-linear cost function); (c) The extension of the neuro-processor concept developed (in this paper) to the case of stochastic shortest path problems; (d) Vehicle routing problems (VRP) under stochastic conditions; and (e) Scheduling problems under nonlinear and/or stochastic conditions. All these problems are extremely challenging and do face a huge computational complexity that can however be, in principle or potentially, significantly alleviated by using and adapting/extending/tuning the neuro-processor based concept developed in this paper. To the best of our knowledge, only very few published works have involved neurocomputing to efficiently address this type of computationally extremely challenging discrete problems. Thus, our neuro-processor concept does contribute to opening the door of neurocomputing for solving challenging discrete optimization problems under difficult settings.

ACKNOWLEDGMENT

The authors are grateful for comments and recommendations from anonymous reviewers who have contributed to improving the scientific quality of this paper.

REFERENCES

[1] S. Lin, "Computer solutions of the traveling salesman problem," *Bell Syst. Tech. J.*, vol. 44, no. 10, pp. 2245–2269, Dec. 1965.

[2] V. Pillac, M. Gendreau, C. Guéret, and A. L. Medaglia, "A review of dynamic vehicle routing problems," *Eur. J. Oper. Res.*, vol. 225, no. 1, pp. 1–11, 2013.

[3] K. Treleaven, M. Pavone, and E. Frazzoli, "Models and efficient algorithms for pickup and delivery problems on roadmaps," in *Proc. IEEE 51st Annu. Conf. Decis. Control (CDC)*, Dec. 2012, pp. 5691–5698.

[4] W. H. Ip, D. Wang, and V. Cho, "Aircraft ground service scheduling problems and their genetic algorithm with hybrid assignment and sequence encoding scheme," *IEEE Syst. J.*, vol. 7, no. 4, pp. 649–657, Dec. 2013.

[5] A. F. Alkaya and E. Duman, "Application of sequence-dependent traveling salesman problem in printed circuit board assembly," *IEEE Trans. Compon., Packag. Manufact. Technol.*, vol. 3, no. 6, pp. 1063–1076, Jun. 2013.

[6] R. Cristescu, B. Beferull-Lozano, M. Vetterli, and R. Wattenhofer, "Network correlated data gathering with explicit communication: NP-completeness and algorithms," *IEEE/ACM Trans. Netw.*, vol. 14, no. 1, pp. 41–54, Feb. 2006.

[7] J. Brożek, "Hybrid algorithm for optimization of m-loop electric power distribution networks," *IEE Proc.-Gener., Transmiss. Distrib.*, vol. 151, no. 2, pp. 246–251, Mar. 2004.

[8] Y. Kato and M. Yasuhara, "Recovery of drawing order from single-stroke handwriting images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 9, pp. 938–949, Sep. 2000.

[9] P. Dennis, "A heuristic routing algorithm," *IEEE Micro*, vol. 3, no. 2, pp. 48–54, Apr. 1983.

[10] C.-H. Cheng, W.-K. Lee, and K.-F. Wong, "A genetic algorithm-based clustering approach for database partitioning," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 32, no. 3, pp. 215–230, Aug. 2002.

[11] M. Niendorf, P. T. Kabamba, and A. R. Girard, "Stability of solutions to classes of traveling salesman problems," *IEEE Trans. Cybern.*, vol. 46, no. 4, pp. 973–985, Apr. 2016.

[12] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Oper. Res.*, vol. 21, no. 2, pp. 498–516, Apr. 1973.

[13] J. W. Pepper, B. L. Golden, and E. A. Wasil, "Solving the traveling salesman problem with annealing-based heuristics: A computational study," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 32, no. 1, pp. 72–77, Aug. 2002.

[14] H.-K. Tsai, J.-M. Yang, Y.-F. Tsai, and C.-Y. Kao, "An evolutionary algorithm for large traveling salesman problems," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 34, no. 4, pp. 1718–1729, Aug. 2004.

[15] H. D. Nguyen, I. Yoshihara, K. Yamamori, and M. Yasunaga, "Implementation of an effective hybrid ga for large-scale traveling salesman problems," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 37, no. 1, pp. 92–99, Feb. 2007.

[16] X.-F. Xie and J. Liu, "Multiagent optimization system for solving the traveling salesman problem (TSP)," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 39, no. 2, pp. 489–502, Apr. 2009.

[17] V. A. Shim, K. C. Tan, and C. Y. Cheong, "A hybrid estimation of distribution algorithm with decomposition for solving the multiobjective multiple traveling salesman problem," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 5, pp. 682–691, Sep. 2012.

[18] L. Ke, Q. Zhang, and R. Battiti, "MOEA/D-ACO: A multiobjective evolutionary algorithm using decomposition and AntColony," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 1845–1859, Dec. 2013.

[19] J. Jackson, M. Faied, and A. Girard, "Comparison of Tabu/2-opt heuristic and optimal tree search method for assignment problems," *Int. J. Robust Nonlinear Control*, vol. 21, no. 12, pp. 1358–1371, Aug. 2011.

[20] F. Pasqualetti, A. Franchi, and F. Bullo, "On cooperative patrolling: Optimal trajectories, complexity analysis, and approximation algorithms," *IEEE Trans. Robot.*, vol. 28, no. 3, pp. 592–606, Jun. 2012.

[21] L. Gouveia and S. Voß, "A classification of formulations for the (time dependent) traveling salesman problem," *Eur. J. Oper. Res.*, vol. 83, no. 1, pp. 69–82, 1995.

[22] J.-C. Picard and M. Queyranne, "The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling," *Operations Res.*, vol. 26, no. 1, pp. 86–110, Feb. 1978.

[23] T. Cheong and C. C. White, "Dynamic traveling salesman problem: Value of real-time traffic information," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 619–630, Jun. 2012.

[24] S. Van Hoesel and A. Wagelmans, "On the complexity of postoptimality analysis of 01 programs," *Discrete Appl. Math.*, vol. 91, nos. 1–3, pp. 251–263, Jan. 1999.

[25] T. Bektas, "The multiple traveling salesman problem: An overview of formulations and solution procedures," *Omega*, vol. 34, no. 3, pp. 209–219, Jun. 2006.

[26] J. Li, M. C. Zhou, Q. Sun, X. Dai, and X. Yu, "Colored traveling salesman problem," *IEEE Trans. Cybern.*, vol. 45, no. 11, pp. 2390–2401, Nov. 2015.

[27] N. R. Mahapatra and S. Dutt, "Scalable global and local hashing strategies for duplicate pruning in parallel A* graph search," *IEEE Trans. Parallel Distrib. Syst.*, vol. 8, no. 7, pp. 738–756, Jul. 1997.

[28] L. Delin, L. Zhang, and L. X. Zhihui, "Heuristic simulated annealing genetic algorithm for traveling salesman problem," in *Proc. 6th Int. Conf. Comput. Sci. Educ. (ICCSE)*, Aug. 2011, pp. 260–264.

[29] A. Plebe and A. M. Anile, "A neural-network-based approach to the double traveling salesman problem," *Neural Comput.*, vol. 14, no. 2, pp. 437–471, Feb. 2002.

[30] G. Faulkner and H. Talhami, “Using ‘biological’ genetic algorithms to solve the travelling salesman problem with applications in medical image processing,” in *Proc. IEEE Int. Conf. Evol. Comput.*, vol. 2, Nov./Dec. 1995, pp. 707–710.

[31] L. Li, S. Ju, and Y. Zhang, “Improved ant colony optimization for the traveling salesman problem,” in *Proc. Int. Conf. Intell. Comput. Technol. Automat. (ICICTA)*, vol. 1, Oct. 2008, pp. 76–80.

[32] J. Zhang and W. Xiong, “An improved particle swarm optimization algorithm and its application for solving traveling salesman problem,” in *Proc. WRI World Congr. Comput. Sci. Inf. Eng.*, vol. 4, Mar./Apr. 2009, pp. 612–616.

[33] J. C. Platt and A. H. Barr, “Constrained differential optimization for neural networks,” Dept. Comput. Sci., California Inst. Technol., Pasadena, CA, USA, Tech. Rep. TR-88-17, Apr. 1988.

[34] E. Wacholder, J. Han, and R. C. Mann, “A neural network algorithm for the multiple traveling salesmen problem,” *Biol. Cybern.*, vol. 61, pp. 11–19, May 1989.

[35] J. J. Hopfield and D. W. Tank, “Neural computation of decision in optimization problems,” *Biol. Cybern.*, vol. 52, pp. 141–152, Jul. 1985.

[36] S. Ghafurian and N. Javadian, “An ant colony algorithm for solving fixed destination multi-depot multiple traveling salesmen problems,” *Appl. Soft Comput.*, vol. 11, no. 1, pp. 1256–1262, Jan. 2011.

[37] J. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI, USA: Univ. Michigan Press, 1975.

[38] Z. Wang, H. Duan, and X. Zhang, “An improved greedy genetic algorithm for solving traveling salesman problem,” in *Proc. 5th Int. Conf. Natural Comput. (ICNC)*, vol. 5, Aug. 2009, pp. 374–378.

[39] M. Max, “Parallel ant colony optimization for the traveling salesman problem,” in *Ant Colony Optimization and Swarm Intelligence*. Berlin, Germany: Springer, 2006, pp. 224–234.

[40] R. Gan, Q. Guo, H. Chang, and Y. Yi, “Improved ant colony optimization algorithm for the traveling salesman problems,” *J. Syst. Eng. Electron.*, vol. 21, no. 2, pp. 329–333, Apr. 2010.

[41] B. Vallade and T. Nakashima, “Improving particle swarm optimization algorithm and its application to physical traveling salesman problems with a dynamic search space,” in *Applied Computing and Information Technology*. Cham, Switzerland: Springer, 2014, pp. 105–119.

[42] A. H. Gee and R. W. Prager, “Limitations of neural networks for solving traveling salesman problems,” *IEEE Trans. Neural Netw.*, vol. 6, no. 1, pp. 280–282, Jan. 1995.

[43] T.-W. Yue and L.-C. Fu, “Ineffectiveness in solving combinatorial optimization problems using a Hopfield network: A new perspective from aliasing effect,” in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 3, Jun. 1990, pp. 787–792.

[44] H. H. Hoos and T. Stützle, “On the empirical scaling of run-time for finding optimal solutions to the travelling salesman problem,” *Eur. J. Oper. Res.*, vol. 238, no. 1, pp. 87–94, 2014.

[45] D. Ben-Arieh, G. Gutin, M. Penn, A. Yeo, and A. Zverovitch, “Transformations of generalized ATSP into ATSP,” *Oper. Res. Lett.*, vol. 31, no. 5, pp. 357–365, Sep. 2003.

[46] Q. Gao and X. Xu, “The analysis and research on computational complexity,” in *Proc. 26th Chin. Control Decis. Conf. (CCDC)*, May 2014, pp. 3467–3472.

[47] J. C. Chedjou and K. Kyamakya, “A novel general and robust method based on NAOP for solving nonlinear ordinary differential equations and partial differential equations by cellular neural networks,” *J. Dyn. Syst., Meas., Control, Trans.*, vol. 135, pp. 031014-1–031014-11, May 2013.

[48] J. C. Chedjou and K. Kyamakya, “A universal concept based on cellular neural networks for ultrafast and flexible solving of differential equations,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 4, pp. 749–762, Apr. 2015.

[49] J. C. Chedjou and K. Kyamakya, “Benchmarking a recurrent neural network based efficient shortest path problem (SPP) solver concept under difficult dynamic parameter settings conditions,” *Neurocomputing*, vol. 196, pp. 175–209, Jul. 2016.

[50] T. Bektaş and L. Gouveiab, “Requiem for the Miller–Tucker–Zemlin subtour elimination constraints,” *Eur. J. Oper. Res.*, vol. 236, pp. 820–832, 2014.

[51] U. Pferschy and R. Stanek, “Generating subtour elimination constraints for the TSP from pure integer solutions,” *Central Eur. J. Oper. Res.*, vol. 25, pp. 231–260, Mar. 2017.

JEAN CHAMBERLAIN CHEDJOU is currently an Associate Professor with the Institute for Smart Systems Technologies, Alpen-Adria University of Klagenfurt, Austria. He is conducting research in the field of dynamic systems in traffic engineering. His current research interests include nonlinear dynamics in intelligent transportation systems (ITS), applications of neural networks and cellular neural networks in ITS, electronics circuits engineering, and graph theory. He has been serving as a Reviewer in several journals, including IEEE ACCESS, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS, the IEEE TRANSACTIONS ON COMMUNICATIONS, NEURO-COMPUTING, NONLINEAR DYNAMICS, SENSORS, the *International Journal of Bifurcation and Chaos*, the *Journal of Applied Physics*, the *AEU - International Journal of Electronics and Communications*.

KYANDOGHERE KYAMAKYA is currently a Full Professor of transportation informatics and the Deputy Director of the Institute for Smart Systems Technologies, University of Klagenfurt, Austria. He is actively conducting research involving modeling, simulation, and test-bed evaluations for a series of concepts in the framework of the application of information and communication technology in transportation. In the research addressing transportation systems, a series of fundamental and theoretical tools from the fields of applied mathematics, electronics, and computer science is either extensively exploited or source of inspiration for innovative solutions and concepts, including nonlinear dynamics and synchronization, cellular neural networks, nonlinear image processing, cellular-neural-network-based analog computing, systems science, and computational intelligence.

NKIEDIEL ALAIN AKWIR received the Ph.D. degree in information and communications engineering from Klagenfurt University, Klagenfurt, Austria. He is currently a Senior Lecturer with the Department of Electrical Engineering, Faculty of Applied Sciences and Technologies, Université Libre des Pays des grands Lacs, Goma, Democratic Republic of the Congo. His research interests include traffic modeling, nonlinear dynamics in transportation, and graph theory.