Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2014, Article ID 108673, 11 pages
http://dx.doi.org/10.1155/2014/108673

Research Article

Threshold Accepting Heuristic for Fair Flow Optimization in Wireless Mesh Networks

Jarosław Hurkała and Tomasz Śliwiński

Institute of Control & Computation Engineering, Warsaw University of Technology, Ulica Nowowiejska 15/19, 00-665 Warsaw, Poland

Correspondence should be addressed to Jarosław Hurkała; j.hurkala@elka.pw.edu.pl

Received 10 February 2014; Revised 27 April 2014; Accepted 7 May 2014; Published 25 May 2014

Academic Editor: Dritan Nace

Developing effective, fairness-preserving optimization algorithms is of considerable importance in systems which serve many users. In this paper we show the results of the threshold accepting procedure applied to extremely difficult problem of fair resource allocation in wireless mesh networks (WMN). The fairness is modeled by allowing preferences with regard to distribution of Internet traffic between network participants. As aggregation operator we utilize weighted ordered weighted averaging (WOWA). In the underlaying optimization problem, the physical medium properties cause strong interference among simultaneously operating node devices, leading to nonlinearities in the mixed-integer pricing subproblem. That is where the threshold accepting procedure is applied. We show that, the threshold accepting heuristic performs much better than the widely utilized simulated annealing algorithm.

1. Introduction

Wireless mesh network (WMN) is an organized cooperating group of network devices communicating with each other by means of wireless media. The nodes are organized in a mesh topology, where each wireless device not only sends and receives its own data but also serves as a relay for other nodes. Some of the nodes can be connected to cable network or mobile network and serve as Internet gateways. This way the whole mesh network constitutes a decentralized way of providing Internet access between attending participants.

This network type poses numerous advantages including setup cost, independence of the hardwired infrastructure, and flexibility. However, providing fair and efficient network management, including routing and scheduling, is not a straightforward task. The main source of difficulty lies in physical medium properties that cause strong interference among simultaneously operating devices. Additionally the link quality is a function of the distance and can be affected by obstacles present between the nodes. As a result the efficient network operation requires transmission scheduling, channel assignment, and transmission power determination.

Common objective of the optimization is maximization of the total throughput while retaining fairness in its distribution between participants.

In network optimization problems fairness is often accomplished using the lexicographic max-min (LMM) optimization. In the case of convex attainable set, this corresponds to the max-min fairness concept [1] which states that in the optimal solution it is impossible to increase any of the outcomes without the decreasing of some smaller (worse) ones [1–3]. In nonconvex case such strictly defined MMF solution may not exist while the LMM always exists and it covers the former if it exists (see [4] for wider discussion). However, LMM is a stiff approach that usually does not allow any other criteria, the overall efficiency (total throughput) in particular. Moreover, it requires sequential repeated optimization of the original problem. A recent survey on fairness oriented WMN optimization can be found in [5].

In the paper a more flexible approach based on the weighted ordered weighted averaging (WOWA) outcomes aggregation is proposed. It provides the consistent, reasonable, and fairness-preserving methodology of modeling various preferences (from the extreme pessimistic through

neutral to extreme optimistic) with regard to distribution of Internet throughput between network participants. It is based on the OWA (ordered weighted aggregation) [6, 7] in which the preference weights are assigned to the ordered values (i.e., to the largest value, the second largest, and so on) rather than to the specific criteria. The WOWA extension allows using additional weights (called importance weights) assigned to the specific outcomes.

The OWA operator provides a parameterized family of aggregation operators, which include many of the well-known operators such as the maximum, the minimum, the k-order statistics, conditional minimax [8] known as conditional value at risk (CVaR) in the field of financial risk measurement, the median, and the arithmetic mean. The OWA satisfies the properties of strict monotonicity, impartiality, and, in the case of monotonic increasing weights, the property of equitability (satisfies the principle of transfers: equitable transfer of an arbitrary small amount from the larger outcome to a smaller outcome results in a more preferred achievement vector). Thus the OWA-based optimization generates the so-called equitably efficient solutions (cf. [9] for the formal axiomatic definition). According to [9, 10], equitable efficiency expresses the concept of fairness, in which all system entities have to be treated equally and in the stochastic problems equitability corresponds to the risk aversion [11]. Since its introduction, the OWA aggregation has been successfully applied to many fields of decision making [7, 12, 13]. When applying the OWA aggregation to multicriteria optimization, the weighting of the ordered outcome values causes the fact that the OWA optimization problem is nonlinear even for linear programming formulation of the original constraints and criteria. Yager has shown that the nature of the nonlinearity introduced by the ordering operation allows one to convert the OWA optimization into a mixed integer programming problem. It was shown [14] that the OWA optimization with monotonic weights can be formed as a standard linear program of higher dimension. Its significant extension introduced by Torra [15] incorporates importance weighting into the OWA operator forming the weighted OWA (WOWA) aggregation as a particular case of Choquet integral using a distorted probability as the measure. The WOWA averaging is defined by two weighting vectors: the preferential weights and the importance weights. It covers both the weighted means (with equal preferential weights) and the OWA averages (with equal importance weights) as special cases. Some of the example applications of importance weights (applied to specific outcomes) include definition of the size or importance of processes in a multiagent environment, setting scenario probability (if uniform objectives represent various possible values of the same uncertain outcome under several scenarios), or job priorities in scheduling problems. It was shown [16] that in the case of monotonic preferential weights WOWA aggregation can also be modeled by a mere linear extension of the original problem.

This paper extends and refines our initial work on the subject presented in [17]. The list- based threshold accepting heuristic applied to the pricing problem is described. The results are compared to the exact MIP formulation.

This paper is organized as follows. In the next section, we present the wireless mesh network optimization problem together with the outline of the solution approach. Next the WOWA aggregation operator is introduced. In the fourth section we deal with the pricing problem and show two alternative approximate solution algorithms. The last section presents the setup and the results of the computational experiments.

2. Flow Optimization in WMN

The WMN networking technology has been drawing an increased attention over the last years (see literature overview in [18, 19] and references therein). Due to complexity of the problem usually some sorts of simplifications are assumed. The problem considered in this paper can be stated as follows. There is given a WMN network with a number of nodes, routers and gateways. The nodes are interconnected wirelessly in compliance with all the physical constraints and requirements, including signal loss with increasing distance and interference occurring during simultaneous operation. Each node can be either sending or receiving data, but not both at the same time. There are a number of modulation and coding schemes (MCSs) used for communication between the nodes with different properties with regard to speed, maximum allowable interference, and the distance. Each MCS has its signal to interference plus noise ratio (SINR) requirement that must be fulfilled in order to successfully transmit data. Only one fixed transmitting power and single channel are assumed, but MCS can be dynamically allocated. The network model consists only of links for which at least one MCS can be applied, and this requirement reduces to the maximum allowable distance between the nodes.

Only downstream communication direction from gateways to routers is considered. For each router there is a single predefined path leading to a chosen gateway. The routers have elastic traffic demand, which means they can consume all the possible network capacity. The demands compete for network resources to get as much throughput as possible.

The objective is to maximize total throughput preserving fairness among competing demands.

The solution approach is based on the concept of compatible sets introduced in [20]. Compatible set consists of links that can operate at the same time within given interference model. The basic solution concept consists in linear approximation of the model and consecutive generation of the compatible sets improving current solution within the column generation schema. The approximation is needed if the time horizon is divided into fixed-length time slots; if not the solution is optimal.

Although we consider only a specific problem, the solution concepts involving application of WOWA operators can be utilized for many other variants of WMN problems including different capacity reservation models (see [19]).

2.1. Notation. Wireless mesh network topology is represented by a directed graph $\mathcal{N} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \mathcal{G} \cup \mathcal{R}$ is the set of nodes from which we distinguish the set of gateways and

set of mesh routers denoted, respectively, by $\mathcal{G}$ and $\mathcal{R}$, and $\mathcal{E}$ is the set of (radio) links.

The (potential) link between two nodes $v, w \in \mathcal{V}$ is modeled by a directed arc $e = (v, w) \in \mathcal{E}$, where $a(e) = v$ is the originating node that can transmit a signal of a given power P_{vw} to its terminating node $b(e) = w$. Additionally, we assume that if arc $e = (v, w) \in \mathcal{E}$ exists, then an opposite arc $e' = (w, v) \in \mathcal{E}$ also exists. Furthermore, the sets of outgoing and incoming arcs from/to node $v \in \mathcal{V}$ are denoted, respectively, by $\delta^+(v)$ and $\delta^-(v)$, while $\delta(v) = \delta^+(v) \cup \delta^-(v)$ is the set of all arcs incident to node v.

Nodes are transmitting using one of the available *modulation and coding schemes* (MCSs) denoted by $m \in \mathcal{M}$, where $\mathcal{M}$ is the set of all MCSs (to simplify the considerations, we assume that the set of available MCSs is $\mathcal{M}(e) = \mathcal{M}, e \in \mathcal{E}$). The (raw) data rate of transmission using MCS m is denoted by B^m.

The (radio) link $e = (v, w)$ can successfully transmit if the signal to noise ratio (SNR) for the arc e is greater than a certain threshold value denoted by γ^m for at least one MCS $m \in \mathcal{M}$:

$$\Gamma'_e = \frac{P_{vw}}{N} \geq \gamma^m, \tag{1}$$

where $N = 10^{-10.1}$ mW is the ambient noise power.

At any arbitrary time instance the transmission of other nodes can interfere with transmission on e. The corresponding signal to interference plus noise ratio (SINR) condition for successful transmission on e using MCS m reads as follows:

$$\Gamma_e = \frac{P_{vw}}{N + \sum_{a \in \mathcal{A} \setminus \{v\}} P_{aw}} \geq \gamma^m, \tag{2}$$

where $\mathcal{A} \subseteq \mathcal{V}$ is the set of active nodes which are transmitting at the same time.

Moreover, we assume that a node can either transmit or receive or be inactive; that is,

$$\begin{aligned} \mathcal{A} \cap \{b(\delta^+(v))\} \neq \emptyset &\Longrightarrow \mathcal{A} \cap \{a(\delta^-(v))\} = \emptyset \quad v \in \mathcal{V}, \\ \mathcal{A} \cap \{a(\delta^-(v))\} \neq \emptyset &\Longrightarrow \mathcal{A} \cap \{b(\delta^+(v))\} = \emptyset \quad v \in \mathcal{V}. \end{aligned} \tag{3}$$

Each router $r \in \mathcal{R}$ is connected with a selected gateway $g \in \mathcal{G}$ by a directed path p_d (i.e., a subset of links, $p_d \subseteq \mathcal{E}$) that is supposed to carry the entire downstream flow f_d from gateway g to router r (to simplify the formulations, we do not consider the upstream direction). The set of routers is considered as demands and denoted by $d \in \mathcal{D}$, where $\mathcal{D} = \mathcal{V} \setminus \mathcal{G}$. Let $\mathcal{P} = \{p_1, \ldots, p_D\}$ be the given set of paths between routers and gateways, where $D = |\mathcal{D}|$. For each link $e \in \mathcal{E}$, the set of all indices of paths in $\mathcal{P}$ that contain this link will be denoted by $\mathcal{Q}_e = \{d : e \in p_d, 1 \leq d \leq D\}$.

2.2. Compatible Sets. A compatible set (CS) is defined as a subset $\mathcal{E}_i$ of links $\mathcal{E}_i \subseteq \mathcal{E}$ together with a particular MCS m_e, $e \in \mathcal{E}_i$ that each link is using so that each link can be active simultaneously (i.e., transmit without generating too much interfering with other links). In other words, a compatible set is defined by $\mathcal{E}_i = \{(e, m) \in \mathcal{E} \times \mathcal{M} : y_e^m = 1\}$, where variables y_e^m form a feasible solution that satisfy (2) and (3).

2.2.1. Master Problem. Using the family of compatible sets denoted by $\mathcal{I}$, the formulation of the max-min flow optimization problem reads as follows:

$$\max f, \tag{4}$$

$$f \leq f_d \quad d \in \mathcal{D}, \tag{5}$$

$$\sum_{d \in \mathcal{Q}_e} f_d \leq c_e \quad e \in \mathcal{E}, \tag{6}$$

$$c_e = \sum_{i \in \mathcal{I}} B_{ei} z_i \quad e \in \mathcal{E}, \tag{7}$$

$$\sum_{i \in \mathcal{I}} z_i = T, \tag{8}$$

$$z \geq 0. \tag{9}$$

In the presented formulation, T is the time of network operation, B_{ei} is the (raw) data rate of a transmission using MCS $m \in \mathcal{M}$ allocated to link $e \in \mathcal{E}$ in compatible set $i \in \mathcal{I}$, that is, either B^m or 0, depending on whether link e is active or not in the compatible set i, and c_e is total amount of data that can be transmitted over link $e \in \mathcal{E}$ in a time interval T.

This formulation is a noncompact, continuous approximation of the MIP problem involving time slots (see [19]); continuous variables z_i define the number of time slots assigned to a compatible set within the time T.

Since $|\mathcal{I}|$ grows exponentially in the network size, the solution is to use the column generation technique [3, 21], where not all the columns of the constraints matrix are stored. Instead, only a subset of the variables (columns) that can be seen as an approximation (restriction) of the original problem is kept. The column generation algorithm iteratively modifies the subset of variables by introducing new variables in a way that improves the current optimal solution. At the end, the set contains all the variables necessary to construct the overall optimal solution which can use all possible columns. New columns are generated in the pricing problem.

2.2.2. Pricing Problem. The pricing problem we consider corresponds to a WMN system in which there are multiple MCSs available and each node can use different MCS in different compatible set. The following formulation is referred to as dynamic allocation of MCSs to nodes:

$$\max \sum_{e \in \mathcal{E}} \pi_e^* B_e, \tag{10}$$

$$X_v = \sum_{m \in \mathcal{M}} x_v^m \quad v \in \mathcal{V}, \tag{11}$$

$$Y_e = \sum_{m \in \mathcal{M}} y_e^m \quad e \in \mathcal{E}, \tag{12}$$

$$\sum_{e\in\delta(v)} Y_e \le 1 \quad v \in \mathscr{V}, \tag{13}$$

$$\sum_{e\in\delta^+(v)} y_e^m = x_v^m \quad v \in \mathscr{V},\ m \in \mathscr{M}, \tag{14}$$

$$u_{ev}^m \ge y_e^m + X_v - 1 \quad v \in \mathscr{V},\ e \in \mathscr{E},\ m \in \mathscr{M}, \tag{15}$$

$$u_{ev}^m \le y_e^m, \quad u_{ev}^m \le X_v \quad v \in \mathscr{V},\ e \in \mathscr{E},\ m \in \mathscr{M}, \tag{16}$$

$$N y_e^m + \sum_{v\in\mathscr{V}\setminus\{a(e)\}} P_{vb(e)} u_{ev}^m \le \frac{1}{\gamma^m} P_{a(e)b(e)} y_e^m \quad e \in \mathscr{E},\ m \in \mathscr{M}, \tag{17}$$

$$B_e = \sum_{m\in\mathscr{M}} B^m y_e^m \quad e \in \mathscr{E}, \tag{18}$$

$$y_e^m \in \{0,1\}. \tag{19}$$

In this formulation, π_e, $e \in \mathscr{E}$, are the current optimal dual variables associated with constraints (6). At each iteration we are interested in generating CS for which the reduced price $\pi \cdot B$ has the biggest and positive value, as we can expect this will improve the current optimal solution as much as possible. The x_v^m is a binary variable indicating whether node v transmits using MCS m, X_v is a binary variable indicating whether node v transmits, y_e^m is a binary variable indicating whether link e is scheduled to be active with the MCS m, Y_e is a binary variable indicating whether link e is active, and B_e is the (raw) data rate of a transmission allocated to link e. Notice that, in this formulation, X_v and Y_e are auxiliary variables and thus either they can be eliminated or their binarity can be skipped. Moreover, observe that applying (2) directly to our model would result in a bilinear constraint. Hence, we have introduced additional (continuous) variables u_{ev}^m to make the constraint (17) linear. This is achieved by adding the constraints (15)-(16); that is, $u_{ev}^m = 1$ if both y_e^m and X_v are equal to 1, and 0, otherwise.

Each node $v \in \mathscr{V}$ and each link $e \in \mathscr{E}$ can use at most one MCS $m \in \mathscr{M}$ in the compatible set (11)-(12). At most one link $e \in \delta(v)$ incident to node v can be active (13) and exactly one link $e \in \delta^+(v)$ outgoing from node v is active and uses MCS m (14), provided the node is active and uses this MCS in the compatible set. The constraints (15)–(17) assure admissible SINR for link e using MCS m in the compatible set. The (raw) data rate B_e of link e in the compatible set is found by (18).

3. Fair Aggregation Operators

As stated before the basic operator used to preserve fairness among outcomes is lexicographic max-min (LMM) which is equivalent to MMF for the linear problems. In such a case it is possible to carry out the MMF procedure based on simple algorithm that in each step uses the dual information to determine the outcomes that are blocked at their highest values possible. In the following steps, only the outcomes are optimized that have not been blocked before (for details see [19]).

On the other hand, in the WOWA aggregation, the original problem is extended by auxiliary constraints and solved in a single step. Let us first introduce the formalization of the OWA operator. In the OWA aggregation of outcomes $\mathbf{y} = (y_1, \ldots, y_n)$ preferential weights $\mathbf{w} = (w_1, w_2, \ldots, w_n)$ are assigned to the ordered values rather than to the specific criteria:

$$A_w = \sum_{i=1}^{n} w_i \theta_i(\mathbf{y}), \tag{20}$$

where $(\theta_1(\mathbf{y}), \theta_2(\mathbf{y}), \ldots, \theta_n(\mathbf{y})) = \Theta(\mathbf{y})$ is the ordering map $R^n \to R^n$ with $\theta_1(\mathbf{y}) \le \theta_2(\mathbf{y}) \le \cdots \le \theta_n(\mathbf{y})$ and there exists a permutation τ of set I such that $\theta_i(\mathbf{y}) = y_{\tau(i)}$ for $i = 1, 2, \ldots, n$.

If the weights are monotonic, $w_1 > w_2 > \cdots > w_{n-1} > w_n$, the OWA aggregation has the property of equitability [14], which guarantees that an equitable transfer of an arbitrarily small amount from the larger outcome to a smaller outcome results in more preferred achievement vector. Every solution maximizing the OWA function is then an equitably efficient solution to the original multiple criteria problem. Moreover, for linear multiple criteria problems every equitably efficient solution can be found as an optimal solution to the OWA aggregation with appropriate weights.

For the maximization problem the OWA objective aggregation can be formulated as linear extension of the original problem, as follows. Let us apply linear cumulative map to the ordered achievement vectors $\Theta(\mathbf{y})$:

$$\overline{\theta}_k(\mathbf{y}) = \sum_{i=1}^{k} \theta_i(\mathbf{y}) \quad k = 1, 2, \ldots n. \tag{21}$$

As stated in [14], for any given vector $\mathbf{y} \in R^n$, the cumulated ordered coefficient $\overline{\theta}_k(\mathbf{y})$ can be found as the optimal value of the following LP problem:

$$\begin{aligned} \overline{\theta}_k(\mathbf{y}) = \max \quad & k t_k - \sum_{i=1}^{n} h_{ki}, \\ \text{s.t.} \quad & t_k - y_i \le h_{ki}, \quad h_{ki} \ge 0 \\ & i = 1, 2, \ldots, n. \end{aligned} \tag{22}$$

The ordered outcomes can be expressed as differences $\theta_i(\mathbf{y}) = \overline{\theta}_i(\mathbf{y}) - \overline{\theta}_{i-1}(\mathbf{y})$ for $i = 2, \ldots, n$ and $\theta_1(\mathbf{y}) = \overline{\theta}_1(\mathbf{y})$. Hence, the maximization of the OWA operator (20) with weights w_i can be expressed in the form

$$\max\left\{\sum_{i=1}^{n} w_i' \overline{\theta}_i(\mathbf{y}) : \mathbf{y} \in Y\right\}, \tag{23}$$

where coefficients w_i' are defined as $w_n' = w_n$ and $w_i' = w_i - w_{i+1}$ for $i = 1, 2, \ldots, n-1$ and Y is the feasible set of outcome vectors $\mathbf{y}$. If the original weights w_i are strictly decreasing, then $w_i' > 0$ for $i = 1, 2, \ldots, n$.

For the WMN flow optimization problem (4)–(9) the final OWA aggregation of the outcomes f_d for all demands/routers can be stated as the following LP model:

$$\max \sum_{k=1}^{D} k w_k' t_k - \sum_{k=1}^{D} \sum_{d=1}^{D} w_k' h_{dk}, \tag{24}$$

subject to

$$h_{dk} \geq t_k - f_d, \quad h_{dk} \geq 0 \quad d,k = 1,2,\ldots,D \qquad \mathbf{f} \in F, \tag{25}$$

where $\mathbf{f} = [f_d]_{d\in\mathcal{D}}$ and F is a feasible set of flows/throughputs defined by (6)–(9).

The OWA aggregation (20) allows modelling various aggregation functions from the minimum ($w_1 = 1, w_i = 0$ for $2,\ldots,n$), through the arithmetic mean ($w_i = 1/n$ for $i = 1,\ldots,n$), to the maximum ($w_n = 1, w_i = 0$ for $i = 1,\ldots,n-1$). However, it is not possible to express the weighted mean. Due to the property of impartiality and neutrality with respect to the individual attributes the OWA aggregation does not allow representing any importance weights allocated to the specific attributes.

The WOWA aggregation is a generalization of the OWA that allows assigning importance weights to specific criteria [22]. In the case of WMN, the importance weights could express the number of end-users hidden behind each router. For example, the importance weight of the router with 5 directly connected users should be 5 times greater than the importance weight of the router with only a single directly connected user.

Let $\mathbf{p} = (p_1,\ldots,p_n)$ be an n-dimensional vector of importance weights such that $p_i \geq 0$ for $i = 1,\ldots,n$ and $\sum_{i=1}^{n} p_i = 1$. The corresponding weighted OWA aggregation of vector $\mathbf{y}$ is defined [15] as follows:

$$A_{w,p} = \sum_{i=1}^{n} \omega_i \theta_i(\mathbf{y}), \tag{26}$$

with

$$\omega_i = w^*\left(\sum_{k\leq i} p_{\tau(k)}\right) - w^*\left(\sum_{k<i} p_{\tau(k)}\right), \tag{27}$$

where w^* is increasing function interpolating points $(i/n, \sum_{k\leq i} w_k)$ together with the point $(0,0)$ and τ representing the ordering permutation for $\mathbf{y}$ (i.e., $y_{\tau(i)} = \theta(\mathbf{y})$). Moreover, function w^* is required to be a straight line when the points can be interpolated in this way. Due to this requirement, the WOWA aggregation covers the standard weighted mean with weights p_i as a special case of equal preference weights ($w_i = 1/n$ for $i = 1,\ldots,n$). Actually, the WOWA operator is a particular case of the Choquet integral using a distorted probability as the measure [23].

Note that such WOWA definition allows us for a clear interpretation of importance weights as the agent (demand) repetitions [24]. Splitting an agent into two agents does not cause any change of the final distribution of outcomes. For theoretical considerations one may assume that the problem can be transformed (disaggregated) to the unweighted one (that means all the agent importance weights are equal to 1); see [22, 25] and examples therein. Thus, the WOWA aggregation with increasing preferential weights is equitable since equally important unit of a smaller outcome is considered with a larger weight.

As shown in [22], maximization of an equitable WOWA aggregation with decreasing preferential weights $w_1 > w_2 > \cdots > w_n$ may also be implemented as the LP expansion of the original problem. In the case of the WMN flow optimization problem (6)–(9), this can be stated as follows:

$$\max \sum_{k=1}^{D} w'_k \left[\frac{k}{n} t_k - \sum_{d=1}^{D} p_d h_{dk}\right], \tag{28}$$

subject to

$$h_{dk} \geq t_k - f_d, \quad h_{dk} \geq 0 \quad d,k = 1,2,\ldots,D, \tag{29}$$

$$\mathbf{f} \in F. \tag{30}$$

If the importance weights are equal to $p_d = 1/D$, the model reduces to the OWA aggregation.

A special case of the generalized WOWA aggregation is defined for single breakpoint and corresponds to optimization of the predefined quantile of the worst outcomes and in finance is known as the CVaR (conditional value at risk). It can be computed as a standard linear extension of the original problem [22]:

$$\max t - \frac{1}{\beta} \sum_{d=1}^{D} p_d h_d \tag{31}$$

subject to

$$h_d \geq t - f_d, \quad h_d \geq 0 \quad d = 1,\ldots,D \tag{32}$$

$$\mathbf{f} \in F. \tag{33}$$

4. Algorithms

4.1. List-Based Threshold Accepting. List-based threshold accepting algorithm (LBTA) is an extension of threshold accepting metaheuristic, which belongs to the randomized search class of algorithms. This rather unknown heuristic has been successfully applied to many difficult problems [26–30]. Since the problem of fair resource allocation in wireless mesh networks is extremely challenging, we have decided to try this underappreciated algorithm.

The search trajectory of LBTA crosses the solution space by moving from one solution to a random neighbor of that solution and so on. Unlike the greedy local search methods which consist of choosing a better solution from the neighborhood of the current solution until such can be found (hill climbing), the threshold accepting allows choosing a worse candidate solution based on a threshold value. In the general concept of the threshold accepting algorithm it is assumed that a set of decreasing threshold values is given before the computation or an initial threshold value and a decrease schedule are specified. The rate at which the values decrease controls the trade-off between diversification (associated with large threshold values) and intensification (small threshold values) of the search. It is immensely difficult to predict how the algorithm will behave when a certain decrease rate is applied for a given problem without running the actual

Require: Initial solution s_1, list size S, set of move operators $m \in M$
(1) $i \leftarrow 0$
(2) **while** $i < N$ **do**
(3) $m \leftarrow \text{random}(M)$
(4) $s_2 \leftarrow m(s_1)$
(5) **if** $C(s_1) \leq C(s_2)$ **then**
(6) $\Delta \leftarrow (C(s_2) - C(s_1))/C(s_1)$
(7) list $\leftarrow$ list $\cup \{\Delta\}$
(8) $i \leftarrow i + 1$
(9) **else**
(10) $s_1 \leftarrow s_2$
(11) **end if**
(12) **end while**
(13) **return** list

Algorithm 1: Creating the list of threshold values.

computation. It is also very common that the algorithm with the same parameters works better for some problem instances and significantly worse for others. These reflections led to the list-based threshold accepting branch of threshold accepting metaheuristic.

In the list-based threshold accepting approach, instead of a predefined set of values, a list is dynamically created during a presolve phase of the algorithm. The list, which in a way contains knowledge about the search space of the underlying problem, is then used to solve it.

4.1.1. Creating the List of Threshold Values. The first phase of the algorithm consists of gathering information about the search space of the problem that is to be solved. From an initial solution a neighbor solution is created using a move function (perturbation operator) chosen at random from a predefined set of functions. If the candidate solution is better than the current one, it is accepted and becomes the current solution. Otherwise, a threshold value is calculated as a relative change between the two solutions,

$$\Delta = \frac{C(s_2) - C(s_1)}{C(s_1)}, \tag{34}$$

and added to the list, where $C(s_i)$ is the objective function value of the solution $s_i \in S$ and S is a set of all feasible solutions. For this formula to work, it is silently assumed that $C : S \rightarrow \mathbb{R}_+ \cup \{0\}$. This procedure is repeated until the specified size of the list is reached. For the algorithm overview see Algorithm 1.

4.1.2. Optimization Procedure. The second phase of the algorithm is the main optimization routine, in which a solution to the problem is found. The algorithm itself is very similar to that of the previous phase. We start from an initial solution, create new solution from the neighborhood of current one using one of the move functions, and compare both solutions. If the candidate solution is better, it becomes the current one. Otherwise, a relative change is calculated. To this point algorithms in both phases are identical. The difference in the optimization procedure is that we compare the threshold value with the largest value from the list. If the new threshold value is larger, then the new solution is discarded. Otherwise, the new threshold value replaces the value from the list, and the candidate solution is accepted for the next iteration. The best solution found during the optimization process is considered final.

The list-based threshold accepting algorithm also incorporates early termination mechanism: after a (specified) number of candidate solutions are subsequently discarded, the optimization is stopped and the best solution found so far is returned.

The optimization procedure of the list-based threshold accepting algorithm is shown in Algorithm 2.

4.2. Simulated Annealing. Simulated annealing (SA) was first introduced by Kirkpatrick et al. [31], while Černý [32] pointed out the analogy between the annealing process of solids and solving combinatorial problems. Applications of the SA algorithm to optimization problems in various fields have been studied [33–36] and to the WMN problem, as well [17].

The optimization process of the simulated annealing algorithm can be described in the following steps. At the start, an initial solution is required. Then, repeatedly, a candidate solution is randomly chosen from the neighborhood of the current solution. If the candidate solution is the same or better than the current one, it is accepted and replaces the current solution. Even if the generated solution is worse than the current one, it still has a chance to be accepted with the so-called acceptance probability. This probability is a function of difference between objective value of the current and the candidate solution and depends on a control parameter taken from the thermodynamics, called temperature (τ). After a number of iterations, the temperature is decreased by a reduce factor (α), and the process continues as described above. The optimization is stopped either after a maximum number of iterations or when a minimum temperature is reached. The best solution found during the annealing process is considered final.

Require: Initial solution s_1, thresholds list L, set of move operators $m \in M$
(1) $i \leftarrow 0$
(2) $s^* \leftarrow s_1$
(3) **while** $i \leq N$ **do**
(4) $m \leftarrow \text{random}(M)$
(5) $s_2 \leftarrow m(s_1)$
(6) $i \leftarrow i + 1$
(7) **if** $C(s_2) \leq C(s_1)$ **then**
(8) **if** $C(s_2) \leq C(s^*)$ **then**
(9) $s^* \leftarrow s_2$
(10) **end if**
(11) $s_1 \leftarrow s_2$
(12) $i = 0$
(13) **else**
(14) $\Delta_{\text{new}} \leftarrow (C(s_2) - C(s_1))/C(s_1)$
(15) **if** $\Delta_{\text{new}} < \max(\text{list})$ **then**
(16) $\text{list} \leftarrow \text{list} \setminus \{\max(\text{list})\}$
(17) $\text{list} \leftarrow \text{list} \cup \{\Delta_{\text{new}}\}$
(18) $s_1 \leftarrow s_2$
(19) $i = 0$
(20) **end if**
(21) **end if**
(22) **end while**
(23) **return** s^*

Algorithm 2: LBTA optimization procedure.

Table 1: Simulated annealing parameters.

Parameter	Description	Value
α	Reduce factor	$1 - \frac{7}{N}$
τ^0	Initial temperature	0.99
δ^0	Minimal difference between solutions	0.01
p^0	Initial acceptance probability	1
N	Number of SA iterations	300000
N_{const}	Number of iterations at constant temperature	10

For the algorithm overview see Algorithm 3, and for the overview of the SA parameters see Table 1.

4.3. Neighborhood Function. The most problem-specific mechanism of both the SA and the LBTA algorithm that always needs a different approach and implementation is the procedure of generating a candidate solution from the neighborhood of the current one, which is called a perturbation scheme, transition operation/operator, or a move function. Although there are many ways to accomplish this task, we have examined the following operators.

(1) Deactivate a node at random.

(2) Activate a random node and select the MCS with the smallest raw rate.

(3) Switch MCS to one with higher raw rate.

(4) Switch MCS to one with lower raw rate.

(5) Switch MCS to a random one.

Require: Initial solution s_1
(1) $s^* \leftarrow s_1$
(2) **for** $i = 1$ **to** N **do**
(3) **for** $t = 1$ **to** N_{const} **do**
(4) $s_2 \leftarrow \textit{perturbate}(s_1)$
(5) $\delta \leftarrow C(s_2) - C(s_1)$
(6) **if** $\delta \leq 0$ or $e^{-\delta/k\tau} > \text{random}(0, 1)$ **then**
(7) $s_1 \leftarrow s_2$
(8) **end if**
(9) **if** $C(s_2) < C(s^*)$ **then**
(10) $s^* \leftarrow s_2$
(11) **end if**
(12) **end for**
(13) $\tau \leftarrow \tau * \alpha$
(14) **end for**
(15) **return** s^*

Algorithm 3: Simulated annealing.

In order to generate a new solution, the LBTA algorithm applies one of the aforementioned operators chosen at random to the current solution. SA on the other hand uses during the whole optimization procedure only one, compound operator, a combination of operators 1, 2, and 5 so that a transition from initial to any feasible solution is possible (see Algorithm 4).

4.4. Implementation Details

4.4.1. Zero Elements. In the first phase of the list-based threshold accepting algorithm the list is populated with

Require: Current solution compatible set CS
Ensure: CS′ = neighbor(CS)
(1) Choose at random $v \in \mathcal{V}$ and $e \in \delta^+(v)$ that satisfy (11)–(14).
(2) **if** $v \in \mathcal{A}$ **then**
(3) **if** random(0, 1) < 1/ |$\mathcal{M}(e)$| **then**
(4) $\mathcal{A} \leftarrow \mathcal{A} \setminus \{v\}$ [deactivate node]
(5) **else**
(6) $m \leftarrow$ random($\mathcal{M}(e) \setminus \{m\}$) [switch MCS]
(7) **end if**
(8) **else**
(9) $\mathcal{A} \leftarrow \mathcal{A} \cup \{v\}$ [activate node]
(10) $m \leftarrow$ random($\mathcal{M}(e)$) [select MCS]
(11) **end if**

Algorithm 4: SA compatible set perturbation scheme.

values of relative change between two solutions $\Delta \geq 0$. After careful consideration, we believe that including zeros in the list is a misconception. In the actual optimization procedure, that is, the second phase, the threshold value is computed only if the new solution is worse than the current one, which means that the calculated relative change will always have a positive value ($\Delta_{\text{new}} > 0$). The new threshold value is compared with the largest value from the list ($T_{h\max}$). Thus, we can distinguish three cases.

(1) $T_{h\max} = 0$: since thresholds are nonnegative from definition, in this case, the list contains all zero elements and it will not change throughout the whole procedure ($T_{h\max}$ is constant). Comparing a positive threshold value Δ_{new} against zero yields in discarding the candidate solution. The conclusions are as follows:
   (a) it does not matter how many zeros are in the list; the effective size of the list is equal to one;
   (b) the algorithm is reduced to hill climbing algorithm that accepts candidate solutions which are at least as good as the current one.

(2) $T_{h\max} > 0$ and $\Delta_{\text{new}} < T_{h\max}$: the largest (positive) threshold value from the list $T_{h\max}$ is replaced by a smaller (positive) threshold value Δ_{new}. The number of zero elements in the list remains the same throughout the whole procedure and therefore is completely irrelevant to the optimization process. The effective list size is equal to the number of positive elements.

(3) $T_{h\max} > 0$ and $\Delta_{\text{new}} \geq T_{h\max}$: the new solution is discarded and the list remains unchanged.

The main idea behind the list is to control the diversification and intensification of the search process. In the early stage of the search, the algorithm should allow covering as much solution space as possible, which means that the thresholds in the list are expected to be large enough to make that happen. In the middle stage, the algorithm should slowly stop fostering the diversification and begin to foster the intensification of the search. In the end stage, the intensification should be the strongest; that is, the list is supposed to contain smaller and smaller threshold values, which induces the discarding of worse solution candidates. As a consequence of that, the algorithm is converging to a local or possibly even a global optimum.

4.4.2. Stopping Criterion. Even though equipped with an early termination mechanism, the LBTA algorithm does not have a solution space independent stopping criterion. If the number of subsequently discarded worse solutions is set too high, the algorithm will run for an unacceptable long time (it has been observed during preliminary tests). Hence, we propose using a global counter of iterations so that when a limit is reached, the algorithm terminates gracefully.

5. Numerical Experiments

The problem defined by the constraints (6)–(9) with the network flows f_d as the optimization criteria was optimized with different aggregation operators: max-min, lexicographic max-min (LMM), CVaR (31)-(32), and WOWA (28)-(29). For the pricing problem (10)–(18) we applied the two approximate methods: the list-based threshold accepting (LBTA) algorithm and the simulated annealing (SA) algorithm and compared them to the exact MIP approach solved using the CPLEX 12.1 optimization package.

The numerical experiments were performed on a number of randomly generated problem instances of different sizes.

The algorithm of generating network topology instances can be described as follows. A grid of length 25 m of 30×30 points is created. Each of the grid points can be chosen to be a mesh router or a mesh gateway. First, the location of each gateway $g \in \mathcal{G}$ is chosen at random. Then, for each router $r \in \mathcal{R}$, a location is chosen at random that satisfies condition (1) for at least one link $e = (g, r)$, $g \in \mathcal{G}$, and MCS $m \in \mathcal{M}$. This condition is equivalent to $d_{gr} \leq d^m$, where d_{gr} is the distance between gateway g and router r and d^m is the maximum distance for selected MCS m. Finally, paths rooted in the gateways are established by iteratively connecting the neighboring routers that are reachable with the highest link rate and, if possible, with the lowest hop count. The specific data for different MCSs are presented in Table 2.

Table 2: IEEE 802.11a MCS, FER 61%, and 1500-byte payload.

MCS m	Raw rate B^m (Mbps)	SINR threshold $\tilde{\gamma}^m$ (dB)	Maximum link length d^m (m)
BPSK 1/2	6	3.5	273.5
BPSK 3/4	9	6.5	230.0
QPSK 1/2	12	6.6	228.0
QPSK 3/4	18	9.5	193.7
16-QAM 1/2	24	12.8	160.2
16-QAM 3/4	36	16.2	131.7
64-QAM 2/3	48	20.3	103.8
64-QAM 3/4	54	22.1	93.5

Although preferential weights determination is an important issue in the theory of ordered weighted averaging [37–39], for the performance check simple generation methodology has been chosen. All the weights, except two, are strictly decreasing numbers with the step 0.1, while the two selected weights ($k = \lfloor n/3 \rfloor$ and $k = \lfloor 2n/3 \rfloor$) differ from the previous ones by 0.5. The importance weights were generated as random values uniformly distributed in the range $[1, 2]$ and then normalized.

For the LBTA algorithm we used the list size of 50000 elements. This value was chosen based on our preliminary tests for selected problem sizes.

To better compare relative performance of the LBTA and SA algorithms, the only stopping criterion for single run was reaching exactly 300000 iterations, the same for all computations and problem sizes. This way we could compare the speed and the convergence per iteration.

All the experiments were performed on the Intel Core i7 3.4 GHz microprocessor using CPLEX 12.1 optimization library for the linear master problem. The results are the average of 10 randomly generated problems of a given size. Computing times are presented in Table 3, optimal objective value in Table 4, and the total number of the columns (compatible sets) generated (with either LBTA or SA) in Table 5. In all tables the test cases for which the timeout of 600 s occurred were marked with a "—" sign.

A note is required on the LMM optimal problem values (Table 4). Here the objective value of the last step of the LMM algorithm is given. That means that if in the previous steps only suboptimal values were reached, in the last step a better (greater) value than in the exact algorithm is possible. That is why the optimal objective values for the LMM should be treated only as a hint to the performance of the algorithm and not as an algorithm absolute quality measure.

The most noticeable advantage of LBTA algorithm over the SA algorithm when applied to the WMN problem is the computing time; in many cases the LBTA is faster than SA by an order of magnitude. The reason for such a good behavior lays not only in the computing speed of LBTA but also in the quality of the results because this affects the number of the generated columns (compatible sets). One can also notice generally better quality of the results, particularly for bigger problems, also compared to the exact MIP model.

Table 3: Computing times (s).

Problem size		Algorithm	Aggregation operator			
$\lvert\mathcal{D}\rvert, \lvert\mathcal{E}\rvert$	$\lvert\mathcal{G}\rvert$		Max-Min	LMM	CVaR	WOWA
50	8	SA	160.8	214.6	116.2	85.2
10	2		9.0	10.1	8.1	5.7
10	4		8.9	10.6	7.9	4.7
10	8		8.4	11.6	7.4	4.3
20	2		34.0	39.0	27.0	22.8
20	4		31.4	40.5	24.8	20.0
20	8		29.8	41.0	27.1	16.4
50	2		205.5	235.8	162.3	145.8
50	4		157.0	188.6	127.3	109.2
10	2	TA	1.0	1.3	1.0	0.9
10	4		1.0	1.3	0.9	0.7
10	8		0.9	1.5	0.9	0.7
20	2		4.4	5.1	3.5	3.9
20	4		4.0	6.3	3.7	3.2
20	8		3.9	6.2	4.0	2.6
50	2		17.1	22.2	16.4	23.1
50	4		11.8	21.3	12.6	16.0
50	8		15.2	21.2	13.3	16.0
10	2	MIP	1.5	1.8	1.5	1.3
10	4		1.2	1.8	0.9	0.7
10	8		1.1	2.0	0.9	0.9
20	2		66.9	86.4	63.0	82.7
20	4		48.2	79.7	42.0	61.7
20	8		63.8	87.7	62.4	64.5
50	2		—	—	—	—
50	4		—	—	—	—
50	8		—	—	—	—

We can also observe that the computation time increases with the number of routers. This is obvious and can easily be explained; as the network grows, the number of variables increases; hence, more compatible sets need to be computed. More interesting thing is that, for the same number of routers, the computation time is the highest for the smallest number of gateways. This can be explained by the fact that when the number of gateways is small, many paths between routers and gateways share the same link, which makes finding (feasible) compatible sets more difficult. This property has especially significant impact on the WOWA aggregation operator, for which the computation time is equal to or decreases with the number of gateways as long as the number of routers is fixed.

6. Conclusion

Effective, general purpose techniques are of considerable importance in many optimization areas. We have shown

Table 4: Optimal problem values (maximization).

Problem size		Algorithm	Aggregation operator			
$\lvert\mathcal{D}\rvert, \lvert\mathcal{E}\rvert$	$\lvert\mathcal{G}\rvert$		Max-Min	LMM	CVaR	WOWA
10	2	SA	2.7	3.2	2.7	29.7
10	4		3.2	8.4	3.2	41.5
10	8		3.6	8.4	3.6	54.0
20	2		1.3	3.4	1.3	39.1
20	4		1.7	9.1	1.7	60.6
20	8		2.2	7.7	2.5	96.0
50	2		0.6	2.6	0.6	90.6
50	4		0.8	3.0	0.8	136.6
50	8		1.2	7.1	1.2	213.9
10	2	TA	2.7	3.2	2.7	29.7
10	4		3.2	8.0	3.2	41.4
10	8		3.6	10.2	3.6	54.0
20	2		1.3	4.2	1.3	38.9
20	4		1.7	14.1	1.7	60.0
20	8		2.0	10.2	2.4	95.0
50	2		0.6	4.5	0.6	93.2
50	4		0.6	6.0	0.7	139.4
50	8		1.2	7.0	1.2	215.9
10	2	MIP	2.8	3.3	2.8	30.2
10	4		3.3	7.9	3.3	42.6
10	8		3.8	7.9	3.8	54.8
20	2		1.4	3.2	1.4	41.2
20	4		1.8	11.2	1.8	63.0
20	8		2.6	5.5	2.7	99.4
50	2		—	—	—	—
50	4		—	—	—	—
50	8		—	—	—	—

Table 5: Number of compatible sets generated.

Problem size		Algorithm	Aggregation operator			
$\lvert\mathcal{D}\rvert, \lvert\mathcal{E}\rvert$	$\lvert\mathcal{G}\rvert$		Max-Min	LMM	CVaR	WOWA
10	2	SA	15.5	17.6	14.3	10.6
10	4		16.5	19.9	15.6	10.4
10	8		16.1	22.8	14.8	9.5
20	2		39.2	47.9	35.8	32.2
20	4		40.9	56.7	35.5	32.5
20	8		43.9	63.3	42.0	28.9
50	2		132.9	161.6	124.6	118.9
50	4		119.0	151.3	109.4	100.8
50	8		136.8	198.8	112.8	90.1
10	2	TA	15.5	18.0	14.6	10.8
10	4		16.2	19.3	14.5	10.3
10	8		14.4	21.5	13.9	9.6
20	2		39.5	44.9	32.5	32.4
20	4		39.6	60.0	36.5	28.8
20	8		35.3	56.0	37.5	27.1
50	2		102.5	133.5	99.5	99.1
50	4		77.5	135.5	80.6	82.0
50	8		104.1	137.2	90.2	83.2
10	2	MIP	19.1	21.0	16.5	12.0
10	4		21.1	24.2	17.0	10.7
10	8		20.1	25.4	16.5	9.9
20	2		52.0	58.2	41.1	37.8
20	4		53.1	64.6	40.3	34.3
20	8		60.3	69.5	49.6	30.6
50	2		—	—	—	—
50	4		—	—	—	—
50	8		—	—	—	—

that one of the most promising algorithms is the list-based threshold accepting metaheuristic. When applied to the pricing problem of WMN optimization it gives a tremendous advantage over the classic and widely utilized simulated annealing algorithm. We have also shown that OWA-based advanced aggregation operators applied to the flow optimization in wireless mesh networks can be effectively modeled and solved when compared to the traditional lexicographic max-min operators.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The research was partially supported by the National Science Centre, Poland, under Grant no. 2011/01/B/ST7/02967 "Integer programming models for joint optimization of link capacity assignment, transmission scheduling, and routing in fair multicommodity flow networks."

References

[1] D. Bertsekas and R. Gallager, *Data Networks*, Prentice Hall, Englewood Cliffs, NJ, USA, 1987.

[2] J. Jaffe, "Bottleneck flow control," *IEEE Transactions on Communications*, vol. 7, pp. 207–237.

[3] M. Pióro and D. Medhi, *Routing, Flow and Capacity Design in Communication and Computer Networks*, Morgan Kaufmann, San Francisco Calif, USA, 2004.

[4] D. Nace and M. Pióro, "Max-min fairness and its applications to routing and load-balancing in communication networks: A tutorial," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 4, pp. 5–17, 2008.

[5] I. Ahmed, A. Mohamed, and H. Alnuweiri, "On the fairness of resource allocation in wireless mesh networks: a survey," *Wireless Networks*, vol. 19, pp. 1451–1468, 2013.

[6] R. R. Yager, "On ordered weighted averaging aggregation operators in multicriteria decisionmaking," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 18, no. 1, pp. 183–190, 1988.

[7] R. R. Yager and J. Kacprzyk, *The Ordered Weighted Averaging Operators, Theory and Applications*, Kluwer Academic, Dordrecht, The Netherlands, 1997.

[8] W. Ogryczak and T. Śliwiński, "On equitable approaches to resource allocation problems: the conditional minimax solutions," *Journal of Telecommunications and Information Technology*, vol. 3, pp. 40–48, 2002.

[9] M. M. Kostreva and W. Ogryczak, "Linear optimization with multiple equitable criteria," *RAIRO Operations Research*, vol. 33, no. 3, pp. 275–297, 1999.

[10] J. Rawls, *The Theory of Justice*, Harvard University Press, Cambridge, Mass, USA, 1971.

[11] D. E. Bell and H. Raiffa, "Risky choice revisited," in *Decision Making Descriptive, Normative and Prescriptive Interactions*, pp. 99–112, Cambridge University Press, Cambridge, UK, 1988.

[12] R. R. Yager and D. P. Filev, *Essentials of Fuzzy Modeling and Control*, Wiley, New York, NY, USA, 1994.

[13] W. Ogryczak, "Multiple criteria linear programming model for portfolio selection," *Annals of Operations Research*, vol. 97, pp. 143–162, 2000.

[14] W. Ogryczak and T. Śliwiński, "On solving linear programs with the ordered weighted averaging objective," *European Journal of Operational Research*, vol. 148, no. 1, pp. 80–91, 2003.

[15] V. Torra, "The weighted OWA operator," *International Journal of Intelligent Systems*, vol. 12, pp. 153–166, 1997.

[16] W. Ogryczak and T. Śliwiński, "On optimization of the importance weighted OWA aggregation of multiple criteria," in *Computational Science and Its Applications—ICCSA 2007*, vol. 4705 of *Lecture Notes in Computer Science*, pp. 804–817, Springer, Berlin, Germany, 2007.

[17] J. Hurkała and T. Śliwiński, "Fair flow optimization with advanced aggregation operators in Wireless Mesh Networks," in *Proceedings of the IEEE Federated Conference on Computer Science and Information Systems*, pp. 415–421, 2012.

[18] I. F. Akyildiz and X. Wang, "A survey on wireless mesh networks," *IEEE Communications Magazine*, vol. 43, pp. 23–30, 2005.

[19] M. Pióro, M. Zotkiewicz, B. Staehle, D. Staehle, and D. Yuan, "On max-min fair flow optimization in wireless mesh networks," *Ad Hoc Networks*, vol. 13, pp. 134–152, 2014.

[20] A. Capone, G. Carello, I. Filippini, S. Gualandi, and F. Malucelli, "Routing, scheduling and channel assignment in WMN networks: optimization, models and algorithms," *Ad Hoc Networks*, vol. 8, pp. 545–563, 2010.

[21] G. B. Dantzig and P. Wolfe, "The decomposition algorithm for linear programming," *Operations Research*, vol. 8, pp. 101–111, 1960.

[22] W. Ogryczak and T. Śliviński, "On efficient WOWA optimization for decision support under risk," *International Journal of Approximate Reasoning*, vol. 50, no. 6, pp. 915–928, 2009.

[23] M. Grabish, S. A. Orlowski, and R. R. Yager, "Fuzzy aggregation of numerical preferences," in *Fuzzy Sets in Decision Analysis, Operations Research and Statistics*, pp. 32–68, Kluwer Academic, Dodrecht, The Netherlands, 1999.

[24] S. J. Brams and A. D. Taylor, *Fair Division: From Cacke Cutting to Dispute Resolution*, Cambridge University Press, Cambridge, UK, 1996.

[25] W. Ogryczak, "On principles of fair resource allocation for importance weighted agents," in *Proceedings of the International Workshop on Social Informatics (SOCINFO '09)*, pp. 57–62, pol, June 2009.

[26] D. S. Lee, V. S. Vassiliadis, and J. M. Park, "A novel threshold accepting meta-heuristic for the job-shop scheduling problem," *Computers and Operations Research*, vol. 31, no. 13, pp. 2199–2213, 2004.

[27] D. S. Lee, V. S. Vassiliadis, and J. M. Park, "List-based threshold-accepting algorithm for zero-wait scheduling of multiproduct batch plants," *Industrial and Engineering Chemistry Research*, vol. 41, no. 25, pp. 6579–6588, 2002.

[28] C. D. Tarantilis and C. T. Kiranoudis, "A list-based threshold accepting method for job shop scheduling problems," *International Journal of Production Economics*, vol. 77, no. 2, pp. 159–171, 2002.

[29] C. D. Tarantilis, C. T. Kiranoudis, and V. S. Vassiliadis, "A list based threshold accepting algorithm for the capacitated vehicle routing problem," *International Journal of Computer Mathematics*, vol. 79, no. 5, pp. 537–553, 2002.

[30] C. D. Tarantilis, C. T. Kiranoudis, and V. S. Vassiliadis, "A list based threshold accepting metaheuristic for the heterogeneous fixed fleet vehicle routing problem," *Journal of the Operational Research Society*, vol. 54, no. 1, pp. 65–71, 2003.

[31] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by simulated annealing," *The American Association for the Advancement of Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[32] V. Černý, "Thermodynamical approach to the traveling salesman problem: an efficient simulation algorithm," *Journal of Optimization Theory and Applications*, vol. 45, no. 1, pp. 41–51, 1985.

[33] C. Koulamas, "A survey of simulated annealing applications to operations research problems," *Omega*, vol. 22, no. 1, pp. 41–56, 1994.

[34] P. Tian, J. Ma, and D.-M. Zhang, "Application of the simulated annealing algorithm to the combinatorial optimization problem with permutation property: an investigation of generation mechanism," *European Journal of Operational Research*, vol. 118, no. 1, pp. 81–94, 1999.

[35] J. Hurkała and A. Hurkała, "Effective Design of the Simulated Annealing Algorithm for the Flowshop Problem with Minimum Makespan Criterion," in *Journal of Telecommunications and Information Technology*, vol. 2, pp. 92–98, 2012.

[36] J. Hurkała and A. Hurkała, "Fair optimization with advanced aggregation operators in a multicriteria facility layout problem," in *Proceedings of the IEEE Federated Conference on Computer Science and Information Systems*, pp. 355–362, 2013.

[37] Y. M. Wang and C. Parkan, "A minimax disparity approach for obtaining OWA operator weights," *Information Sciences*, vol. 175, no. 1-2, pp. 20–29, 2005.

[38] B. S. Ahn, "Preference relation approach for obtaining OWA operators weights," *International Journal of Approximate Reasoning*, vol. 47, no. 2, pp. 166–178, 2008.

[39] G. R. Amin, "Notes on properties of the OWA weights determination model," *Computers and Industrial Engineering*, vol. 52, no. 4, pp. 533–538, 2007.