Computers & Operations Research 39 (2012) 952–963

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

An effective genetic algorithm for network coding ☆

Xiao-Bing Hu *, Mark S. Leeson, Evor L. Hines

School of Engineering, University of Warwick, Coventry, CV4 7AL, UK

ARTICLE INFO

Available online 20 July 2011

Keywords:
Network Coding
Genetic Algorithm
Resource Minimization
Heuristic Rule

ABSTRACT

The network coding problem (NCP), which aims to minimize network coding resources such as nodes and links, is a relatively new application of genetic algorithms (GAs) and hence little work has so far been reported in this area. Most of the existing literature on NCP has concentrated primarily on the static network coding problem (SNCP). There is a common assumption in work to date that a target rate is always achievable at every sink as long as coding is allowed at all nodes. In most real-world networks, such as wireless networks, any link could be disconnected at any time. This implies that every time a change occurs in the network topology, a new target rate must be determined. The SNCP software implementation then has to be re-run to try to optimize the coding based on the new target rate. In contrast, the GA proposed in this paper is designed with the dynamic network coding problem (DNCP) as the major concern. To this end, a more general formulation of the NCP is described. The new NCP model considers not only the minimization of network coding resources but also the maximization of the rate actually achieved at sinks. This is particularly important to the DNCP, where the target rate may become unachievable due to network topology changes. Based on the new NCP model, an effective GA is designed by integrating selected new problem-specific heuristic rules into the evolutionary process in order to better diversify chromosomes. In dynamic environments, the new GA does not need to recalculate target rate and also exhibits some degree of robustness against network topology changes. Comparative experiments on both SNCP and DNCP illustrate the effectiveness of our new model and algorithm.

© 2011 Elsevier Ltd. All rights reserved.

1. Introduction

The notion of coding at the packet level –commonly called network coding –has attracted significant interest since the publication of [1], which showed its utility for multicast in wire-line packet networks. It is now established that network coding may significantly improve network performance in terms of network throughput. It should be noted that conventional network optimization aims to maximize information flow by utilizing as much link capacity as possible, whilst network coding begins with the assumption that full link capacity utilization has already been achieved wherever possible and then attempts to further increase the network throughput at sinks by performing coding at nodes. This advantage of network coding can be understood in the context of the example shown in Fig. 1.(a) and (b). It is assumed in Fig. 1 that two different pieces of one-unit-sized data/packets, α and β, are to be sent from the source node 1 to the sink node 6 and 7, and the capacity of every link is just 1 (in this paper, the link capacity is defined as the same as the one in conventional network optimization [1], all data are one-unit-sized, and all links are of unit-capacity). The aim is to maximize the rate achieved at each sink, i.e., the amount of different data received by a sink at one time. The rate is also measured in data units. Since all links are of unit-capacity, the potential maximal achievable rate at a sink is equivalent to the number of its incoming links. However, different links may carry the same data, such as the two incoming links of node 7 in Fig. 1.(a) and so the actually achieved rate at a sink may be less than the potential rate. This is likely to be the case if the nodes in the network only forward and replicate the data they receive. For example, as illustrated in Fig. 1(a), the sink node 7 can only receive 1 unit of data β at one time, although the other sink node 6 achieves a rate of 2 by receiving both α and β. The information flow in Fig. 1(a) is optimal from the conventional point of view because every link carries one unit of data, which gives a full utilization of the link capacity. However, if node 4 can combine data from its two incoming links through the "+" operation, then by using the "–" operation to decode data, a rate of 2 can be achieved at both sinks, as shown in Fig. 1.(b). It should be noted that the result of operation "+", e.g., $\alpha+\beta$ in Fig. 1.(b), is still an one-unit-sized data element. Therefore, without exceeding

☆A previous version on the preliminary results of this study was presented at CEC2009, Trondheim, Norway, 18th–21st May 2009.

* Corresponding author.
E-mail addresses: Xiaobing.Hu@warwick.ac.uk (X.-B. Hu), Mark.Leeson@warwick.ac.uk (M.S. Leeson), E.L.Hines@warwick.ac.uk (E.L. Hines).

0305-0548/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2011.07.014

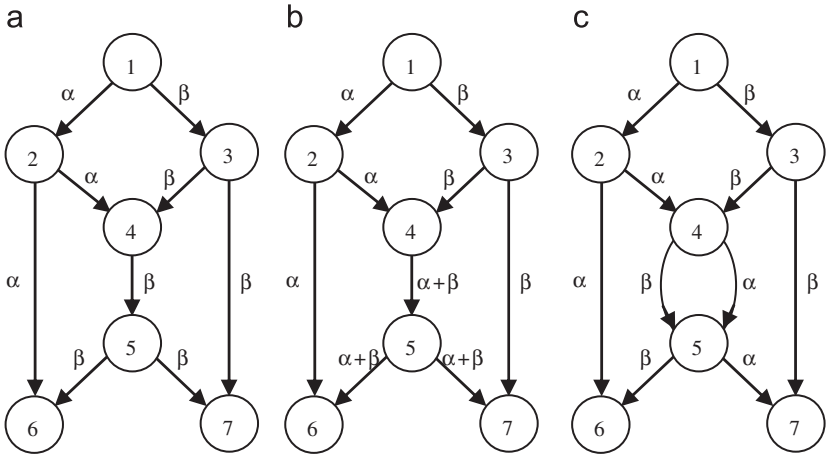


Fig. 1. The basic idea of network coding.

the link capacity, network coding increases the total rate of information flow through the same network from 3 to 4 in Fig. 1, a significant improvement.

Although network coding may be allowed at all nodes in some of the relevant literature, an interesting observation is that a given target rate can often be achieved by conducting network coding at only a relatively small proportion of the nodes [2]. For instance, in the network given by Fig. 1(c), network coding at both nodes 4 and 5 will make no difference in terms of the rate achieved at the sinks or in other word, network coding is not necessary in that network. Therefore, a question is raised: at which nodes does network coding need to be conducted, or how does one make most of network capacity at a minimal cost in terms of network coding resources? To answer this question, a minimal set of nodes needs to be found for coding, and this has been proved to be an NP-hard problem [3]. In this paper, the above problem of minimizing network coding resources is referred to as the Network Coding Problem (NCP). Some attempts have already been made using different methods to address this problem. For instance, two minimal approaches were reported in [4,5] which ascertain the minimal set of nodes for network coding in order to achieve a given target rate. It was determined in [4] that coding is required at no more than d-1 nodes in acyclic networks with 2 unit-rate sources and d sinks. An upper bound on the number of nodes required for both acyclic and cyclic networks was derived in [5]. However, the approaches in both [4,5] determine the minimal set of nodes for coding by removing links in a greedy fashion. A linear programming method was reported in [6] to optimize the various resources used for network coding, and its optimal formulations involved a number of variables and constraints that grow exponentially with the number of sinks.

As large-scale parallel stochastic search and optimization algorithms, genetic algorithms (GAs) have a good provenance in the resolution of diverse NP-hard problems [7,8], including network optimization and resource assignment [9,10]. However, the optimization of network coding is a relatively new area for GAs, and very few results have been reported [2,11–13] to date. The first attempt to apply GAs to network coding was made by Kim et al. [2]. This was then extended from acyclic networks to cyclic networks and from centralized cases to decentralized cases [11]. The genetic representation was the particular focus in later work [12], followed by the proposal of a distributed algorithm to improve GA computational efficiency [13].

There is a common assumption in the network coding optimization work to date that the target rate is always achievable if coding is allowed at all nodes, permitting attention to be focused on the minimization of the number of coding links/nodes. However, in dynamic environments, such as wireless networks, it is quite possible that the target rate may become unachievable due to uncertainty in the connections between the nodes [14]. If this is the case, it is the rate that is actually achieved rather than the target rate that plays a more important role in network coding. The previous studies using the assumption that the target rate was achievable, such as [2,4–13], did not consider the rate actually achieved when minimizing resources, and were therefore largely limited to the static NCP (SNCP). They may only be applied to the dynamic NCP (DNCP) by recalculating the target rate and then re-optimizing resources every time a change occurs in the network topology.

Network coding in dynamic environments is a challenging research topic. Random network coding has proved to be very promising in coping with network topology changes, as it does not require the knowledge of the entire network topology [15–19]. However, in these random network coding studies the minimization of coding resources is not a concern and each node has to communicate its coding weights throughout the network, which means that not all of the network capacity will be available to transmit the signals sent by the source. Therefore, whilst the advantages of random network coding should be acknowledged, it is still necessary to investigate how to improve network coding based on the entire network topology. Robustness against changes in network topology will be a particular concern. As will be explained in Section 3, the DNCP model proposed in this paper exhibits such robustness to some extent when compared with previous work [2,11].

In our previous study [20], preliminary results relating to the design of effective GAs for the minimization of coding resources in the NCP, particularly in the DNCP, were reported but the work was restricted to the NCP utilizing only the simplest addition coding (i.e., the field size is 2) between two incoming signals. The work presented in this paper is concerned with further developments of our previous results by extending our work to NCP where coding with any finite field size is adopted to combine any finite number of incoming signals.

2. Problem formulation

For the sake of simplicity but without losing generality, this paper considers only the one-source-multi-sink NCP, and it is assumed that the source is always node 1 in the network. Let $G(V(t), E(t),t)$ denote the network at time instant t, where $V(t)$ and $E(t)$ are sets of vertices and edges. Suppose G has n_n nodes, n_l

links, and n_s sinks, and R_{Target} is the target rate which is expected to be achieved at every sink.

Before presenting our new NCP model, we would like to give a brief discussion of previous SNCP models. In the SNCP, the target rate R_{Target} is assumed to be achievable if network coding is allowed at all nodes. Then, the SNCP aims to achieve R_{Target} by coding at as few nodes as possible. With network coding at all nodes, the maximum achievable multicast rate is the minimum of the individual maxflow bounds between the source and each of the sinks [1]. An algebraic formulation of the general network coding problem proposed in [21] can be applied to the case where network coding is performed only in some subset of the nodes. Basically, to find the nodes where coding is not necessary, one needs to verify at each potential coding node (PCN, a node with multiple incoming links) whether it is possible to restrict the given node's outputs to depend on a single input without destroying the achievability of the target rate. The above verification can be performed by checking the polynomials of a binary matrix M_y which is constructed based on the coefficients associated with the incoming link(s) of each PCN [2]. A particular M_y is called feasible if the coding scheme defined by it can achieve R_{Target} at every sink. Then the SNCP can be mathematically formulated as the following minimization problem:

$$\min_{M_y} f_S \tag{1}$$

where the objective function f_S is defined as:

$$f_S = \begin{cases} \beta_1 N_{CL} + \beta_2 N_{CN}, & \text{if } M_y \text{ is feasible,} \\ \infty, & \text{if } M_y \text{ is infeasible,} \end{cases} \tag{2}$$

N_{CL} and N_{CN} are the numbers of coding links and nodes, respectively, and β_1 and β_2 are coefficients to adjust the contribution of N_{CN} and N_{CL}, respectively. For instance, in [2], $\beta_1=1$ and $\beta_2=0$.

In the DNCP, the achievability of the initial target rate may not always be guaranteed due to unpredictable changes in network topology arising from signal loss, node failure and the like. Therefore, to apply the SNCP model defined in the Eqs. (1) and (2), to the DNCP, one needs to calculate a new achievable target rate when there is a change in network topology and then completely re-optimize the coding scheme. Clearly, the rate actually achieved at the sinks, which could be a more realistic concern in the DNCP, is not involved in the objective function f_S.

To get rid of the assumption about the achievability of target rate, in this paper, we describe the mathematical model for the NCP as the following maximization problem:

$$\max_{w(i,j,h)} f_D,\ i=1,\ldots,n_n,\ j=1,\ldots,n_{\text{Out}}(i),\ h=1,\ldots,n_{\text{In}}(i), \tag{3}$$

subject to $\text{G}(\text{V}(t),\text{E}(t),t)$,

$$s_{Out}(i,j) = \sum_{h=1}^{n_{In}(i)} w(i,j,h)s_{In}(i,h),\ i=1,\ldots,n_n,\ j=1,\ldots,n_{\text{Out}}(i). \tag{4}$$

$$f_D = \begin{cases} \alpha_1 \min(R(i)) + \alpha_2 ave(R(i)) \\ \quad + \alpha_3/(N_{CL}+1) + \alpha_4/(N_{CN}+1), & \min(R(i)) < R_{Target}, \\ \alpha_1 \min(R(i)) + \alpha_2 ave(R(i)) \\ \quad + \alpha_5/(N_{CL}+1) + \alpha_6/(N_{CN}+1), & \min(R(i)) \geq R_{Target}, \end{cases} \quad i=1,\ldots,n_S, \tag{5}$$

$$\min(\alpha_1,\alpha_2) > \max(\alpha_3,\alpha_4), \tag{6}$$

$$\min(\alpha_5,\alpha_6) \gg \max(\alpha_1,\alpha_2). \tag{7}$$

In the above model, f_D in Eq. (5) is the objective function with α_j, $j=1,\ \ldots\ 6$, as preset coefficients to determine the contribution of different terms in the objective function. $n_{\text{In}}(i)/n_{\text{Out}}(i)$ is the number of incoming/outgoing links of node i. The signals $s_{\text{In}}(i,j)$ and $s_{\text{Out}}(i,j)$ are those on the jth incoming and outgoing links of node i. The weights $w(i,j,h)$, $h=1,\ldots,n_{\text{In}}(i)$, determine how to combine the $n_{\text{In}}(i)$ incoming signals of node i to generate a signal for the jth outgoing link of node i. Actually, Eq. (4) describes the most widely used coding operation: linear network coding. In theory, $w(i,j,h)$ may be continuous but as proved by [15–19], a sufficient number of finite discrete values for $w(i,j,h)$ can guarantee that the maximum possible throughput is achieved. Therefore, in this paper, the value of $w(i,j,h)$ will be chosen from a finite set Θ_W having $N_W \geq 2$ discrete values meaning that the field size for network coding is N_W in this study. Clearly, a set of $w(i,j,h)$ can define how each node in the network forwards, replicates and/or encodes data, in other words, a linear coding scheme is determined by a set of $w(i,j,h)$. Based on the coding scheme determined by $w(i,j,h)$, $R(i)$ is the rate actually achieved at sink i. The optimization of coding scheme, or $w(i,j,h)$, is subject to the network topology defined by $G(V(t), E(t), t)$. Obviously, weights $w(i,j,h)$ are the variables whose values need to be optimized in the NCP. Originally, there should be a set of constraints caused by link capacity. However, because this paper assumes all links are of unit-capacity, therefore link capacity constraints are equivalent to the network topology.

The objective function f_D defined by Eqs. (5)–(7) is more suitable for the DNCP than f_S, in Eq. (2). One can easily see that f_D firstly tries to maximize the overall actually achieved rate, and once the target rate is achieved, the focus of the optimization switches to minimizing the network coding resources. The terms "$\min(R(i))$" and "$\text{ave}(R(i))$" in Eq. (5) can be used to assess the rate that is actually achieved. Basically, a larger term value for "$\text{ave}(R(i))$" is desirable. The optimization of $R(i)$ should be as even as possible to avoid increasing the rate at some sinks by sacrificing that at others. The term representing the value for "$\min(R(i))$" can be used to estimate how evenly $R(i)$ is optimized, i.e., the larger the value is, the more evenly $R(i)$ is optimized. At the same time, as reflected by the terms "$1/(N_{CL}+1)$" and "$1/(N_{CN}+1)$", the network coding resources should be minimized, particularly when the target rate can be achieved, i.e., when $\min(R(i)) \geq R_{\text{Target}}$. Obviously, the coefficients α_1 to α_6 satisfying Conditions (6) and (7) play a crucial role in the automatic switching the focus of optimization from the actually achieved rate to network coding recourses. The above NCP has many considerations for optimization and f_D integrates them into a single weighed objective function. It should be noted that Pareto optimization techniques may be another option for multiple considerations [22], but they are beyond the scope of this work.

Clearly, f_D requires that a solution to the DNCP must make it possible to calculate $R(i)$, N_{CN} and N_{CL}. The binary matrix M_y used in the SNCP model only provides enough information to calculate N_{CN} and N_{CL}. For instance, the modified binary representation in [11] is "at the price of losing the information on the partially active link states that may serve as intermediate steps toward an uncoded transmission state", which means there is not enough information in M_y to calculate the rate $R(i)$ actually achieved at the sinks.

3. Design of GA for DNCP

3.1. The BASIC idea of GAs

Since GAs were introduced based on Darwin's principles by Holland in 1960s, they have been widely used for numerical optimization, combinatorial optimization, classifier systems and many other engineering problems [7,8]. Basically, a GA is a large-scale parallel stochastic searching and optimizing method inspired by the biological mechanisms of natural selection and evolution. Given a population of chromosomes, environmental pressure causes natural selection (survival of the fittest) and

thereby the fitness of the population grows. It is easy to see such a process as optimization. Given an objective function to be maximized, we can randomly create a set of candidate solutions (chromosomes) and use the objective function as an abstract fitness measure (the higher the better). Based on this fitness, some of the better chromosomes are chosen to seed the next generation by applying crossover and/or mutation. Crossover is applied to two selected chromosomes, the so-called parents, and results in one or two new chromosomes, the children. Mutation is applied to one chromosome and results in one new chromosome. Applying crossover and mutation leads to a set of new chromosomes, the offspring. Based on their fitness these offspring compete with old chromosomes for a place in the next generation. This process can be iterated until a solution is found or a previously set time limit is reached. Many components of such an evolutionary process are stochastic. According to Darwin's principles, the emergence of new species, adapted to their environment, is a consequence of the interaction between the survival of the fittest mechanism and undirected variations. Variation operators must be stochastic, the choice of which pieces of information will be exchanged during crossover, as well as the changes in a chromosome during mutation, are random. On the other hand, selection operators can be either deterministic, or stochastic. In the latter case fitter chromosomes have a higher chance to be selected than less fit ones but typically even the weak chromosomes have a chance to become a parent or to survive.

The design of GAs usually includes: choosing an appropriate chromosome structure, developing effective evolutionary operators and introducing useful problem-specific heuristic rules. The following three subsections will describe the new GA developed in our study on the NCP.

3.2. Chromosome structure

Here we need to design a new chromosome structure for the NCP that differs from those in [2,11,12] in order to record the exact information flow on links. The new chromosome structure must make it possible to calculate the actually achieved rate at sinks $R(i)$ for f_D, which is used as the fitness function of the new GA. A straightforward representation is to use the absolute information flow on links to construct chromosomes. However, this representation will cause serious feasibility problems during evolutionary operations,because the set of feasible signals from which a link can choose cannot be predetermined, and varies over time according to the signals on other links. This means that any change in the signal on a link caused by evolutionary operations could make the unchanged signals on some other links infeasible.

Instead of the absolute information flow on links, a chromosome in the new GA records the relative information flow, i.e., an integer k, whose meaning is a certain predefined combination of signals on incoming links of a node. For instance, Fig. 2 shows an illustration of how to predefine k. In this figure, a table is set up to define all possible signal combinations at a node with three incoming links, and the field size is $N_W=3$. A different number of incoming links requires a different predefined table for k, as illustrated in Fig. 3.

Let $head(i)$ denote the serial number of the starting node of link i. It is assumed that the source has as many incoming links as there are signals to be sent, and each signal is associated with one and only one of such assumed links. Let the gene $g(i)$ be associated with link i. Then $g(i)=k$, $k=0, 1, \ldots, N_W^{n_{in}(head(i))}$, where $n_{\mathrm{In}}(head(i))$ is the number of incoming links to node $head(i)$. In other words, for an outgoing link, e.g., link i, the number of possible signal combinations (including no coding) is $N_W^{n_{in}(head(i))}$. The exact combination that a value of k stands for needs to be predefined. Hereafter, the value of $g(i)$ is called the *state* of link i. Then the set of possible *states* for link i is

$$\Theta_S(i)=\{0, 1, \ldots, N_W^{n_{in}(head(i))}\}. \quad (8)$$

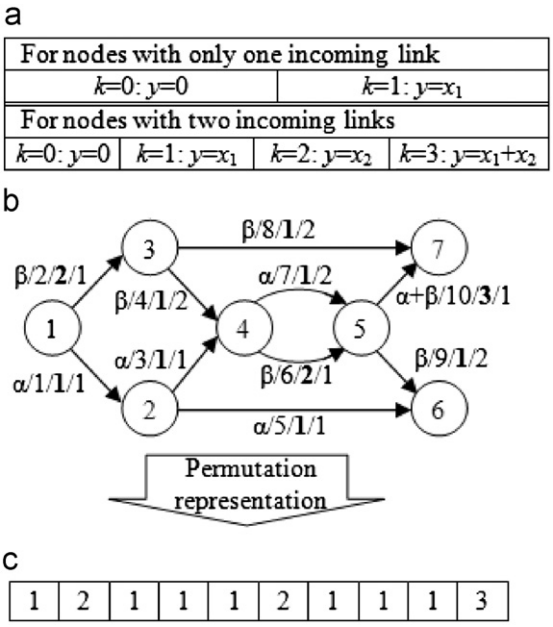
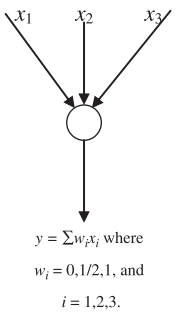


Fig. 3. Chromosome structure based on relative information flow on links. (a) Predefined states of links for the network in Fig. 1b. This field size is set as 2 here for the sake of simplicity. For all, $k=0$: $y=0$. (b) A coding scheme and the resulting information flow (Signal on this link/This link is the ith link in the network/**State of this link (i.e., relative information flow on this link)**/This link is the ith incoming link of the next node). (c) The associated chromosome (In this illustration, the ith gene in the chromosome, i.e., $g(i)=\boldsymbol{k}$, means the signal on the ith link of the network comes from the $\boldsymbol{k}$th incoming link of the previous node; If the previous node has less than $\boldsymbol{k}$ incoming links, then $g(i)=\boldsymbol{k}$ means the ith link is a coding link.



k	y	k	y	k	y
1	$x_1/2$	10	x_1+x_2	19	$x_1/2+x_2/2+x_3/2$
2	x_1	11	$x_1/2+x_3/2$	20	$x_1+x_2/2+x_3/2$
3	$x_2/2$	12	$x_1+x_3/2$	21	$x_1/2+x_2+x_3/2$
4	x_2	13	$x_1/2+x_3$	22	$x_1+x_2+x_3/2$
5	$x_3/2$	14	x_1+x_3	23	$x_1/2+x_2/2+x_3$
6	x_3	15	$x_2/2+x_3/2$	24	$x_1+x_2/2+x_3$
7	$x_1/2+x_2/2$	16	$x_2+x_3/2$	25	$x_1/2+x_2+x_3$
8	$x_1+x_2/2$	17	$x_2/2+x_3$	26	$x_1+x_2+x_3$
9	$x_1/2+x_2$	18	x_2+x_3	0	0

Fig. 2. An illustration of definition of signal combinations.

Therefore, the size of the solution space of the new GA is

$$n_{SP} = \prod_{i=1}^{n_l} N_W^{n_{In}(head(i))}. \quad (9)$$

Unlike the absolute information flow on links, the states in $\Theta_S(i)$ only depend on the network topology and the number of signals that are to be sent, which are both fixed during a GA run. Therefore, as long as $g(i)$ remains within $\Theta_S(i)$ during the evolutionary operation, there will be no feasibility problem. As will be discussed in the following subsection, this condition is very easily fulfilled. On the other hand, the absolute information flow on links can be derived in a straightforward way from a chromosome of the new GA. The simple illustration in Fig. 3 illustrates how to use relative information flow on links to construct a chromosome.

The new chromosome is an integer vector with of size n_l. One may also use an $n_l \times \max(n_{In}(i))$ matrix to record the weights applied to each of the incoming links, and such a matrix representation will need no predefined tables. In this study, we choose the vector representation because: (i) it has a lower memory demand, particularly in the case of large-scale networks and (ii) it is more efficient in terms of algorithm execution (the matrix representation needs to generate $n_{In}(head(i))$ random numbers to determine the relative information flow on link i, whilst the vector representation requires only one random number). However, for networks where a node may have many incoming links, the predefined tables for the vector representation will become unfeasibly large if the field size is also large. For instance, assuming $\max(n_{In}(i))=10$ and $N_W=10$, then the largest predefined table will have 10^{10} entries for k. In this case, we can transform the network into an equivalent network which has a relatively small $\max(n_{In}(i))$, which can actually be just 2, as illustrated in Fig. 4. Then, even if $N_W=100$, the largest predefined table only needs $100^2=10^4$ entries for k. The transformed network will have more links than the original network, which means that, according to Eq. (9), the entire search space will increase, which will become of particular concern in the case of large-scale networks. Therefore, network decomposition methods may be employed here as is the case in many decentralized/distributed algorithms [13].

It should be noted that the search space given by Eqs. (8) and (9) is much larger than those in previous studies. For instance, the search space size for link i is $2^{n_{In}(head(i))}$ in [2], and even down to $n_{In}(head(i))+2$ in [11,12]. Fortunately, as will be explained in Section 4, this disadvantage can be compensated by introducing some new problem-specific heuristic rules based on the exact information flow on links. The new rules would be difficult to apply to previous chromosome structures such as in [11,12] where the exact link information flow is elusive. Actually, as will be shown in the simulation results, the new GA reported in this study can almost always find theoretically optimal solutions, despite the extremely large search space.

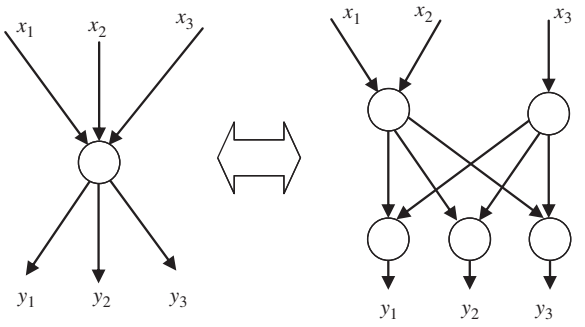


Fig. 4. Transforming a network into a new network with $\max(n_{In}(i))=2$.

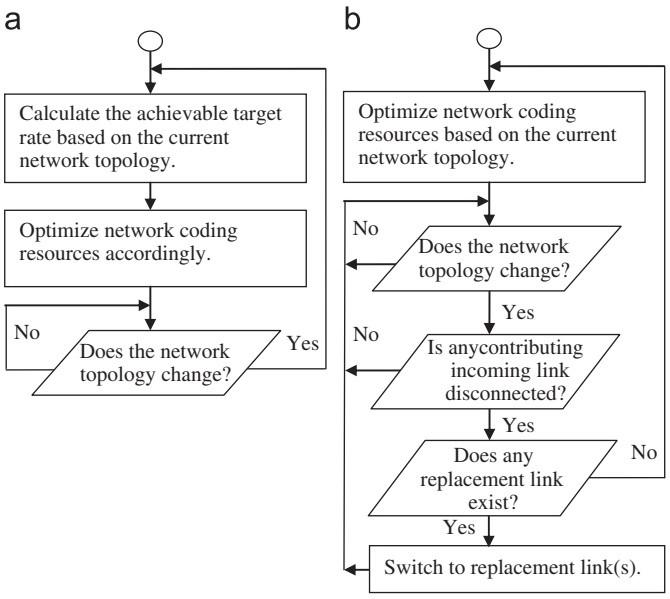


Fig. 5. How to run different GAs in dynamic environments. (a) Apply existing GAs to the DNCP and (b) Apply the new GA to the DNCP.

It should also be noted that the new chromosome structure depends on knowledge of the entire network topology, which may change in the DNCP. Therefore, the new GA cannot achieve the robustness defined by random network coding. Fortunately, the new chromosome structure may still deliver robust performance. This is because a specific coding scheme is defined by a chromosome in the new structure, and it is known whether an incoming link will contribute to a coding instance or not. Therefore, when a non-contributing incoming link is disconnected, there will be no need to re-run the optimization. In other words, a solution based on the new chromosome structure is robust against any changes in non-contributing incoming links. Even if a contributing incoming link is disconnected, one can easily check whether or not there exists any non-contributing incoming link which can replace the disconnected link, because the exact information flow on links are available thanks to the new chromosome structure. For example in Fig. 2, when $k=10$ ($y=x_1+x_2$) performance is not affected by the disconnection of the link carrying x_3. Alternatively, given $x_2=x_3$, then by switching to the link carrying x_3, the performance when $k=10$ is robust against the disconnection of the link carrying x_2. This is impossible for the chromosome structures in [11,12], where coding always involves all incoming links without specifying any of their contributions. Therefore, the new chromosome structure proposed in this paper enables the resulting GA to run in a more efficient and robust way in dynamic environments. As shown in Fig. 5.(a), every time the network topology changes, a re-run after a re-calculation of the achievable target rate is needed for the existing GAs such as those in [2,11,12]. In contrast, for our new GA shown in Fig. 5 (b), there is no need to re-calculate the target rate. Moreover, if only non-contributing incoming links are disconnected or a replacement exists for a disconnected contributing incoming link then there is no need to re-run the optimization. Furthermore, by minimizing the number of coding links, the above robustness associated with the new chromosome structure can be improved because the number of non-contributing incoming links may be increased.

3.3. Evolutionary operators

The mutation operator in this paper is designed as follows. A chromosome is chosen for mutation at a specified probability.

Then a gene associated with a potential coding link needs to be chosen randomly. When gene $g(i)$ is chosen, the associated link is the ith link in the network with a set of possible states given by $\Theta_S(i)$ as defined in Eq. (8). Mutation will randomly choose a value from the set $\Theta_S(i)-\{g(i)\}$, and then reset $g(i)$ to the new value. Thanks to the new chromosome structure, $\Theta_S(i)$ depends only on the network topology, which is fixed during a GA run meaning that the above mutation operation is free of feasibility problems.

This paper adopts uniform crossover, which is highly efficient not only in identifying, inheriting and protecting common genes but also in terms of re-combining uncommon genes [23,24]. Simply speaking, in uniform crossover, each gene of an offspring chromosome inherits the associated gene from its two parent chromosomes with a 50% chance. In the proposed chromosome structure, the ith genes of all chromosomes share the same set ($\Theta_S(i)$) of possible states for link i. Therefore, uniform crossover will cause no feasibility problems. Regarding the choice of two parent chromosomes, any chromosome in an old generation may be chosen as the first parent chromosome at a fixed probability of p_c .Then a different chromosome may be chosen as the second parent chromosome with a probability proportional to its fitness. In this way, every chromosome stands the same chance of becoming the first parent, while a fitter chromosome stands a better chance to cross over with most other chromosomes. This is analogous to a dominant male mating with most females in its territory.

3.4. Heuristic rules

It is well known that heuristic rules, particularly problem-specific ones, often play an important role in the successful applications of GAs. Thanks to the new chromosome structure proposed in Section 3.2, here we can easily integrate the following NCP-specific rules in our new GA.

- Rule 1 All evolutionary operations only apply to the outgoing links of PCNs.
- Rule 2 When initializing the first generation, a certain proportion of chromosomes will allow coding on all PCNs, and for a PCN which has multiple outgoing links, choose at least one link randomly as a coding link. This rule can help to find a solution to achieve the target rate, if it is achievable, at all sinks.
- Rule 3 Furthermore, in the initialization of the first generation, another proportion of chromosomes will allow no coding at all. This rule can help to explore the possibility of maximizing the rate actually achieved at the minimum cost of resources. It should be pointed out that this rule can hardly hit any optimal solution by chance, because even in a network where the optimal solution requires no coding, most no-coding schemes are not optimal.
- Rule 4 In either initialization or evolutionary operations, the states of incoming links of a PCN should be determined in such a way that the node will receive as many different signals as possible. In other words, the signals to a PCN should be diversified as much as possible. This rule will allow as many choices as possible for coding schemes, and therefore can help to diversify a generation.
- Rule 5 For a PCN with multiple outgoing links, there should be a high probability that the outgoing links have different states.
- Rule 6 When initializing the first generation at time instant $t+1$, a certain proportion of chromosomes will inherit the best solution found by the GA at time instant t. This rule is introduced particularly for the DNCP, where, although the network topology changes randomly over time, it is reasonable to assume that there is likely to be a certain consistency in it at two successive time instants. Therefore, inheriting the best solution found at time instant t could be very helpful to the GA in finding a good solution at time instant $t+1$.

It should be noted that the practicability of Rules 4 and 5 relies on the availability of exact information flow on links. Thus the chromosome structures in [2,11–13] do not support these rules as they lack this information but thanks to our new chromosome structure, it becomes possible to integrate them into our algorithm. In the new GA, the following procedure is employed to apply Rule 4 for improving up to one gene in a chromosome. By repeating the procedure, more genes in the chromosome can be improved. A similar procedure is used to apply Rule 5, where outgoing links, rather than incoming links, of PCNs will be considered, and Step 4 will be executed with a specific probability, e.g., 50% in our experiments.

- Step 1 Derive the exact information flow on links which is associated with the chromosome.
- Step 2 Check if there is an unmarked PCN which has at least two incoming links carrying the same signal. If there is no such unmarked PCN, stop.
- Step 3 For the current PCN, among its incoming links which have the same signal, check if there exists at least one link which may receive a signal different from those that the current PCN has already received. If there is no such link, mark the current PCN, and then go to Step 2.
- Step 4 For the current PCN, among its incoming links which have the same signal, randomly choose a link which can bring new signals to the current PCN. Randomly assign such a new signal to that link, modify the associated gene in the chromosome and then stop.

4. Simulation results

4.1. The setup of the experiments

Firstly, the new GA was compared with three existing algorithms for the SNCP, namely the GA reported in [2] (denoted as GA [2]), and two minimal approaches reported in [4,5] (denoted as Minimal 1 and Minimal 2, respectively) to ascertain if the new algorithm can achieve similar or better performance. Then the new GA was tested on the DNCP by applying it to problems where the theoretically optimal solutions are known *a priori* to determine how close the best results generated by the new algorithm are to such theoretically optimal solutions.

In the experiments on the SNCP, there were two sets of test cases, taken from [2] for comparative purposes. The networks in Set I were actually generated by the algorithm in [25], which constructs connected acyclic directed graphs uniformly at random. Two networks with parameters (20 nodes, 80 links, 12 sinks, rate 4) and (40 nodes, 120 links, 12 sinks, rate 3), denoted as SCase 1 and SCase 2, were used for simulations in Set I. The networks in Set II were constructed by cascading a number of copies of network (c) in Fig. 1 such that the source of each subsequent copy of network (c) in Fig. 1 was replaced by an earlier copy's sink. Set II had 4 networks, which used fixed-depth

binary trees containing 3, 7, 15 and 31 copies of network (c) in Fig. 1, respectively. These 4 networks in Set II are referred to as SCase 3 to SCase 6, and they have a maximum multicast rate of 2, which is achievable without coding, i.e., the optimal solutions have no coding links at all. Table 1 provides more details about these SNCP networks, and Fig. 6(a) illustrates how SCase 3 is constructed.

There has been no directly comparable work to this published to date for DNCP algorithms. Therefore, to conduct experiments on the DNCP, appropriate test cases needed to be designed where the theoretically optimal solutions to the DNCP were known *a priori*, so that the performance of the new GA could be precisely assessed. All DNCP test cases in this paper were designed based on Set II of the SNCP networks. In contrast to static networks, where all links remained for the duration of the simulation, in the DNCP test cases, links were introduced that could be disconnected and reconnected dynamically. At each time instant, the dynamic links that were to be disconnected and/or reconnected were randomly chosen. In order to be able to derive the theoretically optimal solutions for a dynamic network at any time instant, only a few constant links in the SNCP networks were replaced by dynamic links in the DNCP test cases. In this study, three categories of link replacements were considered so that the results accounted for different degrees of complexity. Based on the 4 SNCP test cases, 4 DNCP test cases were constructed for each category of replacement. Therefore, there were 12 DNCP test cases, whose main features are summarized in Table 2.

In Category I, for every copy of network (c) in Fig. 1, a permanent link between node 4 and node 5 in network (c) was replaced by a dynamic link, as illustrated in Fig. 6.(b). It can be seen that this category of replacement did not affect the theoretically maximal network throughput (TMNT). This is because, when the dynamic link in a copy of network (c) of Fig. 1 was disconnected, the original TMNT could be achieved by coding at node 4 of that copy. Therefore, in a Category I dynamic network with link replacement, no matter how the dynamic links were disconnected and reconnected, the TMNT remained the same as if all dynamic links were connected, i.e. the theoretical maximal achievable rate (TMAR) at each sink was always 2. The TMNT here can be calculated as $2 \times n_s$. An optimal solution to achieve the TMNT is: coding at node 4 of a copy only if the associated dynamic link is disconnected. In other words, the minimal number of coding nodes/links in the optimal solution, i.e., the theoretically minimal number of coding links (TMNCL) at time instant t, is equal to $n_{\mathrm{DDL}}(t)$, the number of disconnected dynamic links at time instant t.

In Category II, for every copy of network (c) in Fig. 1, the constant links between node 2 and node 6, and node 3 and node 7 in network (c) were replaced by dynamic links, as illustrated in Fig. 6.(c). Clearly, when a dynamic link was disconnected, the TMAR at every one of its downstream sink(s) decreased to 1, regardless of network coding. The TMNT at time instant t can be calculated as $2 \times n_s$-$n_{DS}(t)$, where $n_{DS}(t)$ is the number of downstream sinks of disconnected dynamic links at time instant t. The TMNT can be achieved with no coding, which means the TMNCL is always zero.

Table 1
Networks Used in Different SNCP Test Cases.

	Copy the network Fig. 1.(c) Or generated by [24]	Nodes	Links	Sinks	Target rate
SCase 1	Generated by [24]	20	80	12	4
SCase 2	Generated by [24]	40	120	12	3
SCase 3	3 copies of Fig.1.(c)	19	30	4	2
SCase 4	7 copies of Fig.1.(c)	43	70	8	2
SCase 5	15 copies of Fig.1.(c)	91	150	16	2
SCase 6	31 copies of Fig.1.(c)	187	300	32	2

Table 2
Main Features of 12 DNCP Test Cases.

DNCP Case	Based on which SNCP Case	Category of Link Replacement	Number of Dynamic Links	TMNT	TMNCL
DCase 1	SCase 3	I	3	$2 \times n_s$	$n_{DDL}(t)$
DCase 2	SCase 4	I	7	$2 \times n_s$	$n_{DDL}(t)$
DCase 3	SCase 5	I	15	$2 \times n_s$	$n_{DDL}(t)$
DCase 4	SCase 6	I	31	$2 \times n_s$	$n_{DDL}(t)$
DCase 5	SCase 3	II	6	$2 \times n_s - n_{DS}(t)$	0
DCase 6	SCase 4	II	14	$2 \times n_s - n_{DS}(t)$	0
DCase 7	SCase 5	II	30	$2 \times n_s - n_{DS}(t)$	0
DCase 8	SCase 6	II	62	$2 \times n_s - n_{DS}(t)$	0
DCase 9	SCase 3	III	9	$2 \times n_s - n_{DS}(t)$	$\overline{n}_{DDL}(t)$
DCase 10	SCase 4	III	21	$2 \times n_s - n_{DS}(t)$	$\overline{n}_{DDL}(t)$
DCase 11	SCase 5	III	45	$2 \times n_s - n_{DS}(t)$	$\overline{n}_{DDL}(t)$
DCase 12	SCase 6	III	93	$2 \times n_s - n_{DS}(t)$	$\overline{n}_{DDL}(t)$

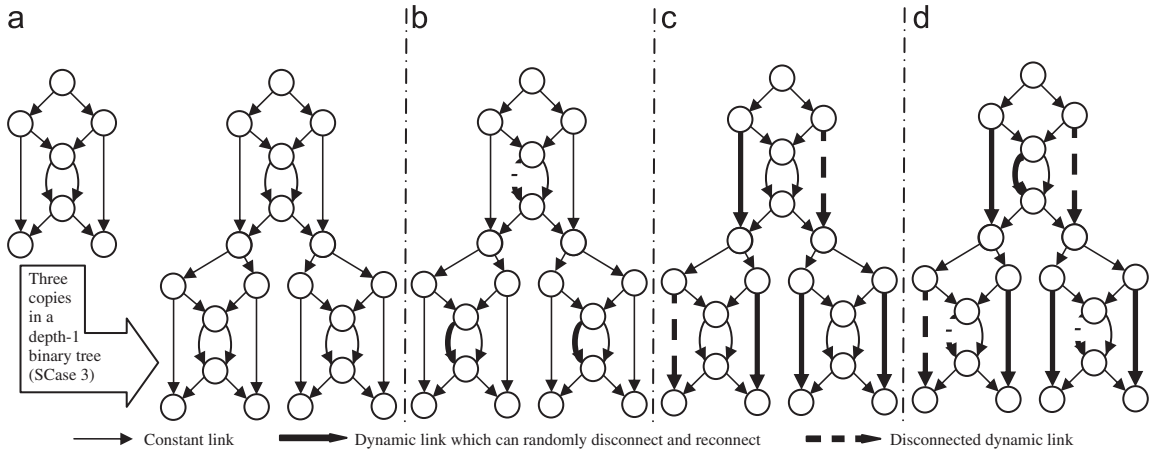


Fig. 6. Some examples of networks used in the experiments. (a) How to construct the network of SCase 3; (b) An example of dynamic networks with link replacement of Category I (DCase 1); (c) An example of dynamic networks with link replacement of Category II (DCase 5); (d) An example of dynamic networks with link replacement of Category III (DCase 9).

Category III was actually a combination of Category I and Category II, as illustrated in Fig. 6.(d). With this category of link replacement, both the TMNT and the TMNCL were dependent on how the dynamic links were disconnected and reconnected. Fortunately, it was quite straightforward to derive the TMNT and the TMNCL of a dynamic network with link replacement of Category III by using the following rules. First, if the dynamic link between node 2 and node 6 (or node 3 and node 7) of a copy was disconnected, then the TMAR of every downstream sink of this disconnected link was 1, and no coding or further analysis needed to be applied to its downstream tree. Second, subject to the first rule, if all downstream sinks of the left/right tree of a copy had a TMAR of 1, then no coding was necessary at node 4 of the copy. Third, subject to the above two rules, if the dynamic link between node 4 and node 5 of a copy was disconnected, an optimal solution required coding at node 4. The TMNT at time instant t was $2\times n_s - n_{DS}(t)$, as in Category II, while the TMNCL at time instant t was equal to $\overline{n}_{DDL}(t)$, the number of disconnected dynamic links as described in Rule 3.

To investigate whether the heuristic rules reported in Section 3.4 really worked, three versions of the new GA were used in the experiments. The first version, denoted as GA1, only employed Rule 1 and Rule 2, the second version, denoted as GA2, used Rules 1, Rule 2 and Rule 3, and the third version, denoted as GA3, adopted Rule 1 to Rule 5. In GA1 to GA3, the mutation probability and crossover probability were 0.1 and 0.4, respectively. The population size and upper bound for the number of generations for evolution were 150 and 300, respectively (the same as in [2]) and the best chromosome of a generation was copied to the next generation directly. In each generation, 15 chromosomes were generated randomly with no link to the previous generation, in order to maintain the diversity in the new generation. The coefficients in the fitness function given by Eq. (5)–(7) were $\alpha_1=\alpha_2=10$, $\alpha_3=1$, $\alpha_4=0$, $\alpha_5=200$ and $\alpha_6=0$. For the sake of simplicity, in most experiments $w(i,j,h)=0$ or 1 in Eq. (4), i.e., the field size was $N_W=2$, unless specified otherwise. In each test case, 20 random experiments were conducted for each algorithm and the network changed randomly 50 times in each DNCP test case. The average results of the experiments follow in the following subsections including statistical analysis (standard deviations (SD) and Kruskal-Wallis tests [26]) for GA1 to GA3. Variations in the results are revealed by the calculation of the former, whilst the latter enables the determination of the significance of the performance differences between GA1, GA2 and GA3. In each Kruskal-Wallis test, the significance of the difference between the performances of GA1 and GA2, GA1 and GA3, and GA2 and GA3 was measured by the value of the occurrence probabilities p_{12}, p_{13} and p_{23} for the respective pairwise comparisons, with a value smaller than 0.05 indicating a statistically significant difference. However, due to the limited space, the SD values are only listed in Tables 4–6; Only Table 7 includes the results of Kruskal-Wallis tests in order to give readers an idea what p_{12}, p_{13} and p_{23} could be like, and in other experiments, the results of Kruskal-Wallis tests are just summarized in the associated discussions.

4.2. The experimental results of SNCP

The average experimental results of the static cases are listed in Tables 3 and 4 reveals more details about the performance of GA1 to GA3. From these results one can make the following observations:

- Table 3 shows that, in the cases of Set I, all the methods perform similarly. In more precise terms, the GAs reported in this paper, i.e., GA1, GA2 and GA3, return slightly lower average numbers of coding links than the existing methods. However, since all methods can find the optimal (i.e., no coding required), or almost optimal solutions to both the cases in Set I, we cannot claim that our new algorithm has a significant advantage when compared with existing algorithms. Analysis of the network topologies in Set I suggests that these networks have too many links; for one network, $n_l=4n_n$, and for the other, $n_l=3n_n$. In the Graph Drawing Community, graphs (i.e., networks) having $n_l=4n_n$ links are actually considered to be dense [25]. In such a network with dense links, it is easy to achieve a relatively small target rate without network coding. Compared with Set I, all the networks in Set II have $n_l<2n_n$. Therefore, although the target rates in Set II are smaller than those in Set I, it is probably more difficult to find a no-coding solution to achieve the smaller target rates in Set II. Actually, in the Set II cases, the results of a comparison of these methods show significant differences, which may suggest that the networks in Set II are more suitable for testing different methods. Therefore, hereafter, we will only focus on analyzing the results of SCase 3 to SCase 6.
- Table 3 also shows that for Set II GA1, GA2 and GA3 clearly outperform the existing algorithms, which struggle to find the theoretically optimal solutions, particularly in complicated cases such as SCase 5 and SCase 6. All three new GAs are capable of finding the theoretically optimal solutions in all 6 SNCP cases, with GA3 always finding the theoretically optimal solutions.
- Since GA1 adopts exactly the same heuristic rules as the GA in [2] but displays superior performance, particularly in SCase 5 and SCase 6 suggesting that the GA designs here are more suitable for the NCP than the GA designs in [2].
- On average, GA2 achieves a better performance than GA1 and the former has one more heuristic rule (Rule 3) it is reasonable to assume that the performance improvement is due to that rule. Moreover, as Rule 3 can also apply to the GA of [2], one my reasonably suppose that the performance of that algorithm would also be improved by Rule 3. However, when one examines the p_{12} in the Kruskal-Wallis test, the values are much greater than 0.05 in each case, so there is no statistically

Table 3
Results of SNCP Simulations (Number of Coding Links).

Methods	Set I				Set II							
	SCase 1		SCase 2		SCase 3		SCase 4		SCase 5		SCase 6	
	Best	Ave	Best	Ave	Best	Ave	Best	Ave	Best	Ave	Best	Ave
Minimal 1	0	1.35	0	1.85	3	3.00	7	7.00	15	15.00	31	31.00
Minimal 2	0	1.85	0	1.90	0	2.15	2	4.70	7	11.60	28	52.80
GA [2]	0	1.20	0	1.05	0	0.65	0	2.15	3	5.35	12	17.20
GA1	0	1.20	0	0.80	0	0.00	0	0.00	0	0.80	0	6.30
GA2	0	1.15	0	0.70	0	0.00	0	0.00	0	0.30	0	5.00
GA3	**0**	**0.00**	**0**	**0.00**	**0**	**0.00**	**0**	**0.00**	**0**	**0.00**	**0**	**0.00**

Table 4
Details of the Results of the new GAs (20 Runs for Each Case).

(Average results of 20 runs)		SCase 1		SCase 2		SCase 3		SCase 4		SCase 5		SCase 6	
		Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Final max fitness	GA1	149.65	45.51	190.00	57.98	240.00	0.00	240.00	0.00	181.67	63.46	40.21	35.06
	GA2	171.54	42.97	195.66	48.30	240.00	0.00	240.00	0.00	195.64	76.24	59.09	29.14
	GA3	**280.00**	**0.00**	**260.00**	**0.00**	**240.00**	**0.00**	**240.00**	**0.00**	**240.00**	**0.00**	**240.00**	**0.00**
How many generations to achieve final max fitness	GA1	245.50	58.21	239.60	115.21	2.35	1.94	9.40	3.78	242.40	85.77	300.00	0.00
	GA2	210.75	53.06	221.00	102.24	1.05	0.24	5.80	10.20	171.90	129.47	300.00	0.00
	GA3	**64.40**	**18.96**	**39.10**	**13.83**	**1.00**	**0.00**	**2.20**	**1.03**	**11.45**	**5.89**	**56.70**	**38.15**
Average minimal coding links	GA1	1.20	0.79	0.80	0.92	0.00	0.00	0.00	0.00	0.80	1.03	6.30	7.09
	GA2	1.15	1.10	0.70	0.48	0.00	0.00	0.00	0.00	0.30	0.67	5.00	3.86
	GA3	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Maximum minimal coding links in all tests	GA1	4		3		0		1		3		22	
	GA2	3		1		0		0		2		12	
	GA3	**0**		**0**		**0**		**0**		**0**		**0**	
Minimum actually achieved rate at sinks in all tests	GA1	3		3		2		2		1		1	
	GA2	3		3		2		2		2		1	
	GA3	**4**		**3**		**2**		**2**		**2**		**2**	
Kruskal-Wallis test results	p_{12}	0.78		0.91		1.00		1.00		0.82		0.62	
	p_{13}	0.00		0.00		1.00		1.00		0.01		0.00	
	P_{23}	0.01		0.01		1.00		1.00		0.02		0.00	

significant difference between the performance of GA1 and GA2 and the advantage of Rule 3 is minor.

- Tables 3 and 4 show that GA3 is the best algorithm of all, its advantages are confirmed by the associated Kruskal-Wallis test results, which are smaller than 0.05 in all static cases except the simplest (SCase 3 and SCase 4) where GA1 and GA2 can often find the optimal solutions. It should be noted that the theoretical maximum fitness is 240 for SCase 3 to SCase 6. Table 6 shows that GA3 always achieves this maximum fitness within 300 generations of evolution (as the associated SD values are zero). From this table, one can see that GA3 converges much more quickly than GA1 and GA2, always finding the theoretical optimal solutions. Since the only difference between GA3 and GA2 is the integration of Rule 4 and Rule 5 into GA3, it is reasonable to conclude that it is the impact of these two additional rules that play a significant role in improving the performance of the GA. As mentioned in Section 3.4, it is the new chromosome structure that makes it possible to integrate these two rules into the new GA. It should be noted that Rule 4 and Rule 5 are not designed solely for the particular networks used in the experiments but are generic and applicable regardless of topology. Therefore, one may say that the new chromosome structure exhibits some advantages as compared with the binary matrix representations used in [2,11–13], where Rule 4 and Rule 5 are not applicable.

4.3. The experimental results of DNCP

With the 12 DNCP test cases, the performances of GA1 to GA3 can be assessed easily by comparing the actually achieved network throughput (AANT) and the actually minimal number of coding links (AMNCL) with the TMNT and the TMNCL. Besides examining the gap between the actually achieved values and the theoretical ones, it is also important to determine how many times the GA has found the optimal solutions, which is reflected by the convergence rate (CR). The closer the CR is to 1, the better the performance of the GA. The experimental results are given in Table 5, from which one can see that:

- In all DNCP test cases, the values actually achieved by GA3 are very close to the theoretical values, and the CR is close to 1. This implies that, in the dynamic environment, GA3 can still converge to the theoretically optimal solutions, regardless of any random changes in the network topology so we can conclude that the GA developed in this paper is very effective at resolving the DNCP.
- In a similar way to the static experiments, GA1 to GA3 have almost the same performance in the simple test cases, i.e., DCase 1, DCase 2, DCase 5, DCase 6, DCase 9 and DCase 10. In the complicated test cases, the overall performance of GA2 is better than that of GA1, i.e., the AANTs of GA2 are generally larger than, or similar to, those of GA1; the AMNCLs of GA2 are smaller than those of GA1, and the CRs of GA2 are larger than those of GA1. There is also a significant increase in performance between GA2 and GA3 in the complicated dynamic test cases. These results again confirm the importance of Rule 3 to Rule 5. In the Kruskal-Wallis tests for DCase 4, DCase 8 and DCase 12, the three most complicated test cases, p_{12} is larger than 0.05, whilst both p_{13} and p_{23} are smaller than 0.05. Therefore, it may be concluded that Rule 3 does not bring any statistically significant improvement, whilst Rule 4 and Rule 5 are highly effective in the DNCP.
- It is worthwhile reminding the reader that the main aim of this DNCP study is to investigate the performance of our GAs in the situation where the target rate may become theoretically unachievable. Therefore any method which is based on the assumption that the target rate is achievable will not be easy to apply. There are many other important issues in the DNCP, such as uncertainties, robustness and time delay that remain to be addressed in future work because the DNCP remains a largely unexplored area in network coding research. Therefore, further extensive investigative effort is required to develop research relating to the DNCP.
- One may notice that in Table 5, the average AMNCL is sometimes smaller than the average TMNCL. This is because the TMNCL is the minimal coding links required to achieve the TMNT, whilst the GAs sometimes converged to a solution which does not achieve the TMNT but uses fewer coding links than the TMNCL.

4.4. Further analysis of rule 4 and rule 5

As has been emphasized throughout this paper, one major contribution of our reported GAs as compared with other existing GAs for the NCP is the new chromosome structure, which makes it possible to calculate the exact information flow on links and

Table 5
Results of the DNCP Tests.

(Ave. results of 20 exp.)		**TMNT**	AANT						**TMNCL**	AMNCL						CR		
			GA1		GA2		**GA3**			GA1		GA2		**GA3**				
			Mean	SD	Mean	SD	Mean	SD		Mean	SD	Mean	SD	Mean	SD			
Link replacement of Category I	DCase 1	**8.00**	8.00	0.00	8.00	0.00	**8.00**	**0.00**	**1.00**	1.00	0.00	1.00	0.00	**1.00**	**0.00**	1.00	1.00	**1.00**
	DCase 2	**16.00**	16.00	0.00	16.00	0.00	**16.00**	**0.00**	**2.00**	2.50	2.63	2.25	2.37	**2.00**	**0.00**	0.75	0.75	**1.00**
	DCase 3	**32.00**	31.00	1.15	31.25	1.50	**31.75**	**0.50**	**4.25**	5.00	4.24	4.50	4.78	**4.00**	**4.08**	0.25	0.50	**0.75**
	DCase 4	**64.00**	55.05	2.45	56.25	5.19	**62.50**	**2.98**	**9.00**	10.25	3.09	7.25	5.25	**6.80**	**7.41**	0.00	0.00	**0.35**
Link replacement of Category II	DCase 5	**7.00**	7.00	0.00	7.00	0.00	**7.00**	**0.00**	**0.00**	0.00	0.00	0.00	0.00	**0.00**	**0.00**	1.00	1.00	**1.00**
	DCase 6	**14.75**	14.75	0.00	14.75	0.00	**14.75**	**0.00**	**0.00**	0.00	0.00	0.00	0.00	**0.00**	**0.00**	1.00	1.00	**1.00**
	DCase 7	**28.50**	28.50	0.00	28.50	0.00	**28.50**	**0.00**	**0.00**	0.00	0.00	0.00	0.00	**0.00**	**0.00**	1.00	1.00	**1.00**
	DCase 8	**58.00**	55.75	4.27	54.00		**58.00**	**0.00**	**0.00**	3.50	0.25	0.75	1.50	**0.00**	**0.00**	0.25	0.30	**1.00**
Link replacement of Category III	DCase 9	**6.25**	6.25	0.00	6.25	0.00	**6.25**	**0.00**	**0.00**	0.00	0.00	0.00	0.00	**0.00**	**0.00**	1.00	1.00	**1.00**
	DCase 10	**12.75**	12.75	0.00	12.75	0.00	**12.75**	**0.00**	**1.00**	1.00	0.00	1.00	0.00	**1.00**	**0.00**	1.00	1.00	**1.00**
	DCase 11	**23.50**	22.35	8.01	23.50	0.00	**23.50**	**0.00**	**0.25**	3.25	4.05	0.25	0.00	**0.25**	**0.00**	0.70	1.00	**1.00**
	DCase 12	**46.25**	43.25	14.45	44.50	12.41	**46.05**	**0.0**	**0.45**	4.25	4.85	6.30	10.01	**1.05**	**1.53**	0.20	0.25	**0.75**

Table 6
Computational efficiencies of different GAs based on SCASE 6.

(Ave. results of 20 exp.)	GA1	GA2	GA3 with a N_{R4R5} of									
			1	2	3	4	5	6	7	8	9	10
Final max fitness	40.21	59.09	**240.00**	**240.00**	**240.00**	**240.00**	**240.00**	**240.00**	**240.00**	**240.00**	**240.00**	**240.00**
Number of coding links	6.30	5.00	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
Generations to converge	300.00	300.00	56.70	28.40	15.60	11.10	4.30	3.90	3.40	3.00	3.00	**2.70**
Computational time of one generation (sec.)	1.39	**1.38**	2.92	3.42	4.16	4.85	5.78	6.48	7.09	7.28	7.82	8.72
Total computational time (sec.)	415.70	414.57	168.31	96.78	60.87	49.91	24.91	25.36	24.18	**22.00**	23.45	23.83

therefore facilitates the integration of new problem-specific heuristic rules, particularly Rule 4 and Rule 5 into the system. To further verify this contribution, the roles played by Rule 4 and Rule 5 will be examined in more detail in this subsection. For brevity, the experimental results reported here are all based on only one case, i.e., SCase 6, which is the most difficult static case. In all previous experiments, when the focus was to improve a chromosome, GA3 applied Rule 4 and Rule 5 no more than once. In other words, GA3 used Rule 4 and Rule 5 to modify no more than one gene of a chromosome. In this subsection, GA3 is allowed to apply Rule 4 and Rule 5 to modify up to N_{R4R5} genes of a chromosome, where $N_{R4R5}=1, \ldots, 10$. All other algorithm related parameters remain the same as in previous experiments.

The results are given in Table 6, from which the following observations can be made. GA3 can always find the theoretical optimal solutions, while GA1 and GA2 often struggle to do so and so we state that Rule 4 and Rule 5 are the cause of the advantages and that applying them more times leads to better performance since the using a larger N_{R4R5} means that GA3 needs fewer generations to converge. However, applying these rules causes an additional computational burden producing longer computation times for a generation of GA3 than for GA1 and GA2. Fortunately, when we combine the computational time consumed by a generation and the generations needed to converge to the optimal solutions, it becomes clear that the total computational time consumed by GA3 to find the optimal solutions is actually smaller than those of GA1 and GA2. Considering the influence of N_{R4R5} on the total computational time of GA3, it seems a balance should be made to set up N_{R4R5} because the smallest total computational time occurs with a medium sized N_{R4R5}. In the case of SCase 6, the best value for N_{R4R5} is 8, which results in GA3 being able to find the optimal solutions at the fastest speed.

Hence it may be concluded that GA3 outperforms GA1 and GA2 in terms of not only the quality of the solution but also in terms of computational efficiency. This shows that the introduction of Rule 4 and Rule 5 is very advantageous and hence justifies the use of the new chromosome structure as proposed in this paper.

4.5. The influence of field size on the performance of our new GAs

It is well known that a large enough field size plays a crucial role in achieving the maximum possible throughput. However, Eq. (8) shows that, in the case of our new GAs, the search space size for a single outgoing link will grow exponentially with the field size, which implies that a larger field size might make it more difficult for our new GAs to find even good solutions. Therefore, the focus of this subsection is to explore and examine the influence of field size on the performance of our new GAs. Five field sizes, $N_W=2$, 4, 6, 8 and 10, were used in GA1, GA2 and GA3 with N_{R4R5} set to 8 for GA3 based on the above. To maintain this work at a reasonable length, the networks used in this experiment were those static cases in Set II, and only the average results for the final max fitness and total computational time are given in Table 7.

From the table, several observations can be made. The field size has a significant influence on the performance of GA1. In the case of SCase 3, the simplest network of all, GA1 with a different field size can always find the optimal solutions but it takes more time when a larger N_W is adopted. In the case of SCase 4 and SCase 5, GA1 may still find the optimal solutions when N_W is small, but the solution quality reduces quickly as N_W increases. In SCase 6, the most complex network of all, GA1 struggles and usually can only find feasible solutions, regardless of the value of N_W. It can be seen in Table 7 that the influence of field size on the performance of GA2 is similar to its influence on GA1. The only difference is that GA2 performs a little better than GA1 in the simple cases. In the case of SCase 6, GA2 has almost the same performance as that

Table 7
The influence of field size on the performance of new GAs.

(Ave. results of 20 exp.)			N_W									
			2		4		6		8		10	
			Mean	SD	**Mean**	SD	**Mean**	SD	**Mean**	SD	**Mean**	SD
GA1	SCase 3	Final max fitness	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00
		Total computational time (sec.)	**0.48**	0.03	**0.81**	0.05	**1.06**	0.07	**3.44**	0.11	**5.56**	0.18
	SCase 4	Final max fitness	**240.00**	0.00	**240.00**	0.00	**123.33**	53.41	**115.68**	58.02	**90.61**	35.29
		Total computational time (sec.)	**2.40**	0.08	**59.26**	2.31	**88.65**	7.34	**93.46**	6.65	**89.27**	7.12
	SCase 5	Final max fitness	**181.67**	63.46	**54.09**	32.71	**50.05**	36.04	**49.72**	28.69	**52.18**	33.60
		Total computational time (sec.)	**158.85**	20.75	**200.61**	23.82	**201.12**	19.46	**200.25**	20.92	**200.42**	21.44
	SCase 6	Final max fitness	**40.21**	35.06	**45.33**	30.38	**42.95**	28.25	**39.76**	24.74	**41.03**	25.97
		Total computational time (sec.)	**415.70**	54.81	**420.58**	49.93	**405.17**	51.26	**411.54**	53.48	**425.67**	55.83
GA2	SCase 3	Final max fitness	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00
		Total computational time (sec.)	**0.47**	0.02	**0.47**	0.03	**0.50**	0.03	**0.48**	0.03	**0.47**	0.03
	SCase 4	Final max fitness	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00
		Total computational time (sec.)	**2.76**	0.10	**6.88**	0.19	**7.85**	0.20	**6.55**	0.20	**8.43**	0.22
	SCase 5	Final max fitness	**195.64**	76.24	**142.86**	46.86	**45.96**	34.16	**47.45**	29.05	**46.11**	32.33
		Total computational time (sec.)	**127.24**	16.38	**201.60**	22.75	**199.53**	24.04	**200.97**	23.60	**204.34**	19.74
	SCase 6	Final max fitness	**59.09**	29.14	**40.16**	37.36	**39.50**	28.74	**41.88**	33.06	**42.02**	30.94
		Total computational time (sec.)	**414.57**	52.70	**429.06**	50.32	**418.63**	54.64	**424.41**	50.96	**419.24**	53.46
GA3	SCase 3	Final max fitness	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00
		Total computational time (sec.)	**2.12**	0.09	**2.05**	0.07	**1.97**	0.07	**2.16**	0.08	**2.01**	0.07
	SCase 4	Final max fitness	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00
		Total computational time (sec.)	**3.37**	0.11	**4.15**	0.13	**3.68**	0.10	**3.44**	0.09	**3.51**	0.13
	SCase 5	Final max fitness	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00
		Total computational time (sec.)	**5.36**	0.19	**5.72**	0.21	**6.44**	0.23	**5.51**	0.20	**7.70**	0.23
	SCase 6	Final max fitness	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00	**240.00**	0.00
		Total computational time (sec.)	**22.00**	1.07	**19.24**	1.03	**21.27**	1.06	**24.84**	1.11	**19.13**	1.05

of GA1, regardless of the value of N_W. In all test cases, in terms of either solution quality or computational time, GA3 has a very robust performance against the change of N_W. Actually, for a given network, GA3 can always find the optimal solution with similar computational time, no matter what value N_W has. In the associated Kruskal-Wallis tests, p_{12} is usually larger than 0.05 (except in SCase 4 with $N_W \geq 6$), which means the performances of GA1 and GA2 are statistically the same; p_{13} is smaller than 0.05 in SCase 4 with $N_W \geq 6$, and in SCase 5 and SCase 6 for all N_W values, and p_{23} is smaller than 0.05 in SCase 5 and SCase 6, which implies GA3 is statistically more robust than GA1 and GA2. In summary, one can see that the field size has a significant influence on GA1 and GA2, which have relatively poor local-searching capability but, due to Rule 4 and Rule 5, no obvious influence on GA3 is observed. This implies that the proposed GA, with its new chromosome structure and consequently the new heuristic rules, can perform satisfactorily for different field sizes.

Based on the test cases for GA3 in Table 7, where N_W makes no difference in the performance of GA3, one may ask: How important is field size for exact network coding? In fact, the importance of field size is mainly appreciated in random network coding because a larger field size means a higher probability of achieving the target rate when random coding is used [15–17]. However, even for a small field size, say $N_W=2$, there could still exist a coding scheme to achieve the target rate. For instance, in a rectangular grid network where the source sends out two signals using the random coding scheme, the probability that a node located at grid position (x,y) relative to the source can decode both signals is at least $(1-1/N_W)^{2(x+y-2)}$ [16]. This implies that for all $N_W \geq 2$, a rate of 2 is in theory always achievable at any node in a finite grid network. Unfortunately, the probability is so small under a small field size, say $N_W=2$, it is improbable that random coding can determine a correct coding scheme. By employing a powerful method of searching, such as GA3, exact network coding may still stand a good chance of finding a correct coding scheme; even when $N_W=2$. In other words, the field size might not be as important to exact network coding as is it to random network coding. This is definitely an issue worth further investigation in future research.

5. Conclusions and future work

As a relatively new information theory, network coding has already demonstrated a significant influence on many research areas such as communication systems, network protocol, wireless networks, and network security. The optimization of network coding, which aims to minimize network coding resources such as coding nodes and links, has recently attracted research attention, with efforts to date focused mainly on the static network coding problem (SNCP). This paper has considered how to address the dynamic network coding problem (DNCP) by proposing its general formulation, and then developing an effective Genetic Algorithm (GA) to realize it. The new model proposed in this paper discards the popular assumption that a target rate is always achievable as long as all nodes allow coding, and introduces the actually achieved rate at sinks, along with the target rate and required resources, to the objective function in order to optimize the system. In order to derive the rate actually achieved, the GA reported is designed based on a new chromosome structure, which allows each chromosome to record a specific network protocol and coding scheme. As a result, the information flow on links can be checked at any stage of optimization. The new representation also makes evolutionary operations free of feasibility problems, and makes it easy to integrate some useful problem-specific heuristic rules into the algorithm. The effectiveness of the new model and GA is illustrated by simulations of both the SNCP and the DNCP.

There are several directions for future research so that the NCP models and the GA reported in this paper may be improved and/or extended. For example, since the proposed GA takes into account the entire network topology its scalability may be poor in

large-scale networks. Therefore, it is necessary to develop distributed/decentralized versions of the proposed GA. This can be done partially by subdividing network coding into the problem of choosing an appropriate coding subgraph (flow-based or queue length based approaches) and characterizing the throughput of network coding along with random and deterministic code constructions.

Furthermore, the DNCP is a major concern of this study but there are still many important issues in dynamic environments to be addressed. For instance, the proposed GA exhibits robustness to some extent against certain changes in network topology but more theoretical work needs to be done to reveal such robustness in more depth, and effective methods/guidelines should be developed to improve such robustness in the future.

The simulation results show that the field size might not be so important in the case of exact network coding as it is to random network coding. However, further effort is needed in order to explore this issue more thoroughly. In specific terms, it could be very useful to develop some results which can achieve the target rate by minimizing the field size together with coding resources.

Although the networks adopted in the simulation were widely used as benchmark problems in previous studies, they are relatively simple and thus a crucial next step is to test the proposed GA, probably in a modified form, on some real-world large-scale networks, such as wireless sensor networks.

Acknowledgements

This work was supported by EPSRC Grant EP/F033591/1. The authors would also like to thank the anonymous reviewers, whose comments were of great help to improve the quality of the paper.

Appendix: Definitions of terms and notations

AANT Actually Achieved Network Throughput
AMNCL Actually Minimal Number of Coding Links
CR Convergence Rate
DNCP Dynamic Network Coding Problem
GA Genetic Algorithm
NCP Network Coding Problem
PCN Potential Coding Node
SNCP Static Network Coding Problem
TMAR Theoretical Maximal Achievable Rate
TMNCL Theoretically Minimal Number of Coding Links
TMNT Theoretically Maximal Network Throughput
f_D/f_S The objective function in the DNCP/SNCP model.
$g(i)$ The gene associated with link i.
k State of a link.
$n_{In}(i)/n_{Out}(i)$ Number of incoming/outgoing links of node i.
$n_n/n_l/n_s$ Number of nodes/links/sinks in a network.
$s_{In}(i,j)/s_{Out}(i,j)$ Signal on the jth incoming/outgoing link of node i.
$w(i,j,h)$, $h=1, \ldots, n_{In}(i)$ Weights determining linear coding for the jth outgoing link of node i.
$G(V(t), E(t), t)$ A network at time instant t, where $V(t)$ and $E(t)$ are sets of vertices and edges.
M_y Binary matrix in the SNCP model.
N_{CN}/N_{CL} Number of coding nodes/links.
N_{R4R5} Number of genes that Rule 4 and Rule 5 are allowed to modify in a chromosome.
N_W Field size for network coding.
$R(i)$ Actually achieved rate at sink i.
R_{Target} The target rate which is expected to be achieved at every sink.
α_j, $j=1, \ldots 6$ Coefficients in f_D.
β_1 and β_2 Coefficients in f_S.
$\Theta_S(i)$ The set of possible *states* for link i.
Θ_W Finite set for the value of $w(i,j,h)$.

References

[1] Ahlswede RN, Cai SYR, Li, Yeung RW. Network information flow. IEEE Trans Inform Theory 2000;46(4):1204–16.
[2] Kim M, Ahn CW, Medard M, Effros M. On minimizing network coding resources: An evolutionary approach, NetCod 2006, Boston, 7 April 2006.
[3] Richey MB, Parker RG. On multiple steiner subgraph problems. Networks 1986;16(4):423–38.
[4] Fragouli C, Parker RG. Information flow decomposition for network coding. IEEE Trans Inform Theory 2006;52(3):829–48.
[5] Langberg M, Sprintson A, Bruck J. The encoding complexity of network coding. IEEE Trans Inform Theory 2006;52(6):2386–97.
[6] Bhattad K, Ratnakar N, Koetter R, Narayanan KR. Minimal network coding for multicast, in: Proceedings IEEE ISIT'05, pp. 1730–4, Adelaide, SA, 4–9 Sep, 2005.
[7] Holland JH. Adaptation in Natural and Artificial Systems. Ann Arbor, MI: University of Michigan Press; 1975.
[8] Eiben AE, Smith JE. Introduction to Evolutionary Computing. Berlin, Germany: Springer-Verlag; 2003.
[9] Rose C, Yates RD. Genetic algorithms and call admission to telecommunications networks. Computers & Operations Research 1996;23(5):485–99.
[10] Mendes JJM, Gonçalves JF, Resende MGC. A random key based genetic algorithm for the resource constrained project scheduling problem. Computers & Operations Research 2009;36(1):92–109.
[11] Kim M, Medard M, Aggarwal V, O'Reilly UM, Kim W, Ahn CW, Effros M. Evolutionary approachs to minimizing network coding resources, Proceedings of the 26th Annual IEEE Conference on Computer Communications (INFOCOM 2007), Anchorage, AK, May 2007.
[12] Kim M, Aggarwal V, O'Reilly UM, Medard M, Kim W. Genetic representation for evolutionary minimization of network coding resources, In: Proceedings of the 4th European Workshop on the Application of Nature-Inspired Techniques to Telecommunication Networks and Other Connected Systems (EvoCOMNET 2007), Valencia, Spain, April 2007.
[13] Kim M, Aggarwal V, O'Reilly UM, Medard M. A Doubly Distributed Genetic Algorithm for Network Coding, In: Proceedings of the 9th annual conference on Genetic and evolutionary computation (GECCO 2007), London, England, 2007.
[14] Ho T, Leong B, Medard M, Koetter R, Chang Y, Effros M. The Utility of Network Coding in Dynamic Environments. In: IWWAN, 2004.
[15] Ho T, Médard M, Shi J, Effros M, Karger DR. On randomized network coding, in 41st Annual Allerton Conference on Communication, Control and Computing, Monticello, USA, 2003.
[16] Ho T, Koetter R, Medard M, Karger DR, Effros M. The benefits of coding over routing in a randomized setting, in: IEEE International Symposium Information Theory, Yokohama, Japan, 2003.
[17] Ho T, Médard M, Koetter R, Karger DR, Effros M, Jun S, Leong B. A Random Linear Network Coding Approach to Multicast. IEEE Trans Inform Theory 2006;52(10):4413–30.
[18] Cooper C. On the distribution of rank of a random matrix over a finite field. Random Struct Algorithms 2000;17:197–212.
[19] Campo AT, Grant A. On Random Network Coding for Multicast, available online.
[20] Hu XB, Leeson MS, Hines EL. An Effective Genetic Algorithm for the Network Coding Problem, The 2009 IEEE Congress on Evolutionary Computation (CEC2009), 18–21 May 2009, Trondheim, Norway.
[21] Koetter R, Medard M. An algebraic approach to network coding. IEEE/ACM Transactions of Networking 2003;11(5):782–95.
[22] Ngatchou P, Anahita Zarei, El-Sharkawi MA. Pareto Multi-Objective Optimization, Proceedings of the 13th International Conference on Intelligent Systems Application to Power Systems, pp. 84–91, 6–10 Nov. 2005.
[23] Sywerda G. Uniform crossover in genetic algorithms, Proceedings of the 3rd International Conference on Genetic Algorithms, pp. 2–9, San Francisco, USA.
[24] Falkenauer E. The worth of uniform crossover, Proceedings of the 1999 Congress on Evolutionary Computation, pp. 782, CEC99, USA.
[25] Melancon G, Philppe F. Generating connected acyclic digraphs uniformly at random. Inf Process Lett 2004;90(4):209–13.
[26] Corder GW, Foreman DI. Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach. New York: Wiley; 2009.