African Journal for Industrial and Applied Mathematics
Afr. J. Ind. Appl. Math., Vol. 1 , No. 1 , pp. 141-157

DOI: 10.4208/ajiam.2025-0007
June 2025

The Genetic Algorithm for Interoperability Cost Optimization: A Case Study of GIMAC Business Network

Eyenga Ovono Tatiana*

University of Yaounde I, BP 176 Yaoundé, Cameroon

Received 2 January 2025; Accepted (in revised version) 25 April 2025

Abstract. Interoperability cost analysis is a continuous trend in enterprise information systems. Researchers have made the most significant contributions in this field. The problem we address is to show how to optimize interoperability costs within a networked enterprise. To overcome this issue, we have used a genetic algorithm in which, the fitness function that we have defined is a new model for optimizing the interoperability cost. To show how organizations validate and verify the solutions obtained through interoperability cost analysis and what measures are taken to ensure the accuracy and reliability of the results, we have conducted a case study within a networked enterprise. The results of the interoperability cost analysis show that the inter-organizational business process studied is not adapted to the interoperable information system and does not allow the evolution of the network after a certain number of years.

AMS subject dassifications: 05C50, 05C78, 65K10

Keywords: Genetic algorithm, interoperability cost analysis, model for optimizing interoperability cost, inter-organizational business process.

1 Introduction

Interoperability in socio-technical systems describes the ability of discrete and technically or organizationally heterogeneous systems to share services or resources with other systems [2]. The interoperability of information systems is defined as the ability of two or more information systems to interact based on multiple understandable requests initiated by useful web services which perform tasks of inter-organizational business processes, to achieve common goals defined by the networked enterprises. This collaboration is putting in place through an interoperable information system. With the rapid growth of industry, enterprises need to collaborate to improve their

*Corresponding author. *Email:* tatiana.eyenga@facsciences-uy1.cm (E. O. Tatiana)

benefits and conquer new markets. This is done through the establishment of a set of partnerships for creating a networked enterprise. During the running time of the networked enterprises information systems for executing an inter-organizational business process, a lot of resources are consumed. This consumption of resources has a cost named: the overall cost of interoperability [3]. We consider that the overall cost of interoperability of information systems represents the total consumption of resources when the interoperation mechanism is carried out at the level of web services, which perform the tasks of an inter-organizational business processes in a network of enterprises, despite the environmental risks associated with their information systems. To improve the interoperability of information systems of networked enterprises in order to add a new partner to the network, the overall interoperability cost must be optimized for establishing and maintaining inter-enterprises collaboration. The optimization of the overall cost of interoperability is the issue that we intend to address in this paper. For that, in the second section we provide a set of basic definitions to clearly understand our work. Section 3 gives the problem description and the mathematical model. In Section 4, we describe the CostGA optimizer based on a genetic algorithm for interoperability cost optimization. In Section 5, we define the procedure of the program that runs inside the optimizer. Section 6: We conduct our case study, and we conclude our paper by the list of perspectives of our work.

2 Definitions

Genetic algorithm (GA) is an optimization algorithm that is inspired by natural selection and is a population-based search algorithm that utilizes the concept of survival of fittest [7]. The general principle behind genetic algorithms is to randomly select a population of individuals from a solution space. This population serves as candidate solutions for optimizing the given problem. The individuals in this population are evaluated using a fitness function. A selection mechanism is used to choose the individuals that will serve as parents for the next generation. These individuals are crossed and mutated to form an offspring. Finally, the next generation is formed by a learning mechanism combining the parent and child individuals. This procedure is repeated until a stopping condition is satisfied. In the next section, we provide the genetic algorithm basic concepts and the encoding schemes for most optimization problems.

2.1 Genetic algorithm concepts

Referring to the fact that a genetic algorithm is an algorithm inspired by the natural sciences, it is important to give a clear understanding of the concepts used like population, chromosome, individual, and gene, for solving a problem in another field different from computational science. So, a population is defined as a set of N individuals representing solutions to a given problem. Each individual can be represented by one or more chromosomes. When an individual is represented by a single chromosome,

the individual merges with the chromosome. A chromosome is one of the solutions to the problem, and each chromosome is made up of a set of genes. The genes correspond to the variables to be optimized or decision variables, whose values are called alleles. In genetic algorithms, we have many techniques for initializing the population, namely: randomness, compositionality, and generality. For more information about population initialization techniques for evolutionary algorithms, the reader can be referred to [8]. In this paper, we will use generality population initialization technique.

2.2 Encoding schemes

The encoding schemes serve to represent an individual in a genetic algorithm. The information of an individual can be represented by a bit or string according to the problem domain. We distinguish several encoding schemes: real, binary, octal, hexadecimal, permutation, value-based, and tree [7]. In this paper, we are interested in real encoding schemes. A real encoding scheme helps to design genetic algorithms whose chromosomes are vectors of floating-point numbers and whose alleles are real numbers [11]. Real parameter vectors are used in our proposed genetic algorithm. Each chromosome in a population is represented by a vector. A vector consists of a set of decision variables. These decision variables are the web services implementing the tasks of an inter-organizational business process. Thus, for each instance of an inter-organizational business process execution corresponds a service utility vector at the end of the execution. After multiple business process executions, we have multiple business process instances, which also correspond to multiple service utility vectors. Therefore, we build an interoperability cost optimizer that, for these multiple service utility vectors, identifies the best vector, i.e. the components of this vector where the services have the best service utility percentages, and the overall interoperability cost of the business processes which is the final scalar obtained at the end of the procedure, is minimized. Let us provide the mathematical description of the problem of interoperability cost analysis in which decision variables can be identified.

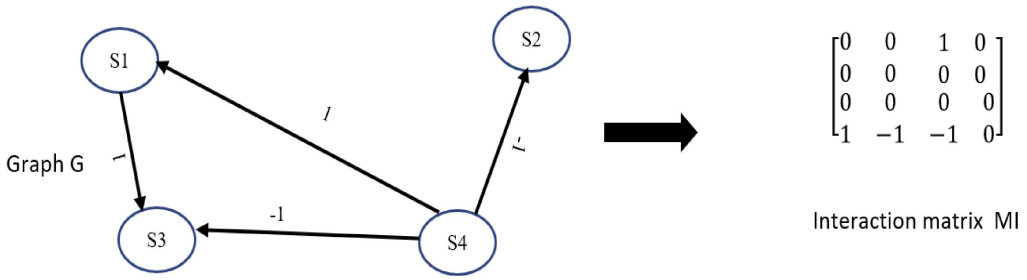
3 Problem description and mathematical model

The problem of analyzing the overall cost of interoperability aims to determine the set of useful services needed to execute business process instances in networked enterprise information systems. More significantly, the aim is to determine the set of services useful for executing inter-organizational business process instances, knowing that the aim is to minimize exchanges between services and resource consumption. The set of services executing the instances of an inter-organizational business process can be represented in the form of a dependency graph between services. Services will be represented as nodes in a dynamic weighted directed graph. A web service is linked to other services via its ports. A web service receives messages through its input ports and sends messages through its output ports. The orientation of the arc

indicates the direction of the message flow exchanged, and the weighting of the arc reflects the usefulness or otherwise of the information exchanged for the target service. Thus, depending on the type of interaction between services when exchanging messages and resources, we distinguish three types of interaction as we can see in Fig. 1: those based on usable (exploitable) message exchanges, encoded by the number 1, i.e. a successful interaction; those based on non-usable exchanges, i.e. a failure, encoded by the number -1; and the absence of interaction, encoded by the number 0.

Let $MI \in \{-1, 0, 1\}^{\theta \times \theta}$ be the interaction matrix derived from the dependency graph between services executing business process instance tasks in interoperable networked enterprise information systems after one time step. MI_{ij} represent the type of interaction between service i and service j. Fig. 1 represents this mechanism. For each interaction between two services, an unforeseen event may occur in the information system, disrupting data transmission on either the source or target service side.These unforeseen events can include organizational contingencies (power cuts, low bandwidth), security contingencies (network attacks disrupting service operation), and organizational culture and governance structures, which have an impact on the overall cost of interoperability. In this paper, we have materialized these risks using the column vector R of size θ. θ represents the order of the dependency graph between services $G = (V, E)$. An element R_i of R denotes the probability of the interruption associated with service i; $R_i \in [0; 1]$. For the sake of simplicity and in order to avoid privileging one disruption over another, the different elements of this vector will be equal to 0.5.

Problem description: The problem of analyzing the total cost of interoperability can be defined as follows: let us consider θ interacting services of inter-organizational business processes of an interoperable information system of networked enterprises. Each service $j \in \{1, 2, 3, \ldots, \theta\}$ is subject to a perturbation R_j associated with the information system environment and has an interoperability cost C_j defined by its response time and throughput.



-Web service S1 and S3: successful interaction; Web service S2 and S4 : failure interaction; Web service S1 and S2 : absence of interaction. Web services S1, S2, S3 and S4 carry out the tasks of an inter-organizational business process, interacting with each other by exchanging messages and data. These interactions are represented by the dependency graph between services G, whose computational representation is the interaction matrix between services MI.

Figure 1: A dependency graph between services and interaction matrix.

- Objects to be analyzed: Interaction matrices between services MI of size $\theta \times \theta$.
- Question: How to determine the services of inter-organizational business process instances with the best utility percentages U that minimize the overall cost of interoperability $fval$?
- Solution: Model the overall interoperability cost analysis problem as a constrained nonlinear optimization problem. The solution to this problem is defined by the interoperability cost optimization model as follows:

$$\min f(U)_{U\in\Re^{\theta}} = \sum_{j=1}^{\theta} \mathrm{C}_j \times \mathrm{U}_j \tag{3.1}$$

$$sc \begin{cases} \sum_{j=1}^{\theta} \mathrm{MI}_{ij} \times \mathrm{U}_j \geq \mathrm{R}_i, \\ 0 \leq U \leq 1, \\ i, j \in \{1, 2, 3, \ldots, \theta\}. \end{cases}$$

3.1 Literal translation of the mathematical model and definition of model properties

- $\min f(U)_{U\in\Re^{\theta}} = \sum_{j=1}^{\theta} \mathrm{C}_j \times \mathrm{U}_j$: This means that we want to find a service utility vector U that minimizes the overall cost of interoperability.
- $\sum_{j=1}^{\theta} \mathrm{MI}_{ij} \times \mathrm{U}_j \geq \mathrm{R}_i;\ i, j \in \{1, 2, 3, \ldots, \theta\}$: This constraint means that the disruptions induced by the information systems environment must be less than the interactions between the services of inter-organizational business processes instances.
- $0 \leq U \leq 1$. The utility percentage of the service can be 0% or 100%, so the vector's components are bounded between 0 and 1.
- U: Column vector, representing the service utility vector,dimension θ.
- f: $\Re \to \Re$, the objective function.
- C_j components of service interoperability cost vector. This value can be calculated using the QoS attributes; in our work, we only use response time and throughput.
- If $\mathrm{U}_j \in [0, 50[$, the service j is not useful for business process interoperability.
- If $\mathrm{U}_j \in [50, 80[$, the service j is partially useful for business process interoperability.
- If $\mathrm{U}_j \in [80, 100]$, the service j is completely useful for business process interoperability.
- The $fval$ value of the objective function represents the overall cost of interoperability.

3.2 Calculation of service interoperability cost vector components

We calculate the components of service interoperability cost vector C using QoS attributes that emphasize exchanges between services: response time (Re) and throughput (Th). Re_j and Th_j represents respectively the response time and the throughput of the service j. We have: $\mathrm{Re}_j \in [\mathrm{Re}_0; \mathrm{Re}_{\max}]$ and $\mathrm{Th}_j \in [\mathrm{Th}_0; \mathrm{Th}_{\max}]$ represents the theoretical minimum response time that we aim to achieve, which improves the business process interoperability performance. $\mathrm{Re}_{\max}$ represents the theoretical maximum response time that degrades the business process interoperability performance. Th_0 represents the theoretical minimum flow rate or throughput that degrades the business process interoperability performance. $\mathrm{Th}_{\max}$ represents the theoretical maximum throughput that we aim to achieve, which improves the business process interoperability performance. To compensate for the different units of measurement between the different QoS attribute values, the values need to be normalized to lie within the interval $[0, 1]$; for example, response time should be normalized by minimization, while throughput should be normalized by maximization [1]. The interoperability cost C_j of the service j is given by the formula

$$\mathrm{C}_j = \left(\frac{\mathrm{Vmax}_{\mathrm{Re}} - \mathrm{Re}_j}{\mathrm{Vmax}_{\mathrm{Re}} - \mathrm{Vmin}_{\mathrm{Re}}} + \frac{\mathrm{Th}_j - \mathrm{Vmin}_{\mathrm{Th}}}{\mathrm{Vmax}_{\mathrm{Th}} - \mathrm{Vmin}_{\mathrm{Th}}} \right), \tag{3.2}$$

where $\mathrm{V\,max}_{\mathrm{Re}}$ represents the maximum response time of all services participating in the interoperability mechanism. $\mathrm{V\,min}_{\mathrm{Re}}$ represents the minimum response time of all services participating in the interoperability mechanism. $\mathrm{Vmax}_{\mathrm{Th}}$ represents the maximum throughput value of all services participating in the interoperability mechanism. $\mathrm{Vmin}_{\mathrm{Th}}$ represents the minimum throughput value for all services participating in the interoperability mechanism. θ: is the number of services. Re_j represents the effective response time of service j. Th_j represents the effective throughput of service j. Finally, the steps taken to determine the overall cost of the interoperability are:

- Transform the dependency graph between services into a matrix, which we have named the service interaction matrix.
- Define the risks associated with the interoperable information system environment that could disrupt exchanges between the web services implementing the business process tasks like a vector R.
- Calculate the interoperability cost associated with each service C_j (see Eq. (3.2)), taking into account response time and throughput, which are service quality attributes.
- Take into consideration the components of the service utility vector U, which are the problem variables.
- Determine the solution of Eq. (3.1), which is the extreme vector U^* such that $f(U^*) \leq f(U)$ for any $U \in R^\theta$. So, it is the optimal solution of the problem $f(U^*)$ that gives the optimal value $fval$, $f(U^*) = fval$, which in our work represents the overall cost of interoperability.

4 The CostGA optimizer

When we decide to use a genetic algorithm as a linear program that would run inside of an optimizer, it is important to know what we want the genetic algorithm to do. To answer this question, the literature review offers us two points of view on the use of genetic algorithms: on the one hand, for design problems and, on the other, for repetitive problems. As mentioned by the authors [6] , in design problems, the genetic algorithm is used to develop an optimizer to find the right solution. In this case, we have many trials, and it is not a worry if some executions end up with bad results. In repetitive problems, on the other hand, the genetic algorithm is an optimizer in its own right, and in this case, we aim to reduce the probability of obtaining bad results by carrying out a single execution very often. In this scenario, the algorithm can then be used later in different problem instances to compare different versions of this algorithm experimentally, taking into account their performance measures. In our work, we present the practical aspect of using the genetic algorithm to develop the CostGA optimizer used to solve the global interoperability cost optimization problem. In the next section, we present the detailed description of the algorithm.

4.1 Code scheme and the fitness

Real coding scheme is used in CostGA. Each chromosome in the population represent the service utility vector. In the code of a solution $U_{ti} = \{U_{ti,1}, U_{ti,2}, \ldots, U_{ti,\theta}\}$, θ represent the number of web services implementing the inter-organizational business tasks. $U_{ti,j} \in [0\%; 100\%[= [0; 1[$, $j \in [1, \theta]$ refers to the utility percentage of the service j. So that U_{ti} is a vector where each component represents the utility percentage of a service when the i-th-instance of the inter-organizational business process is executed. For example, the service with identifier $\theta - 1$ in the U_{t1} vector is said to be useful for the interoperability of business process instances. Fig. 2 represents an example of solution. Initial solutions are randomly generated according to the encod-

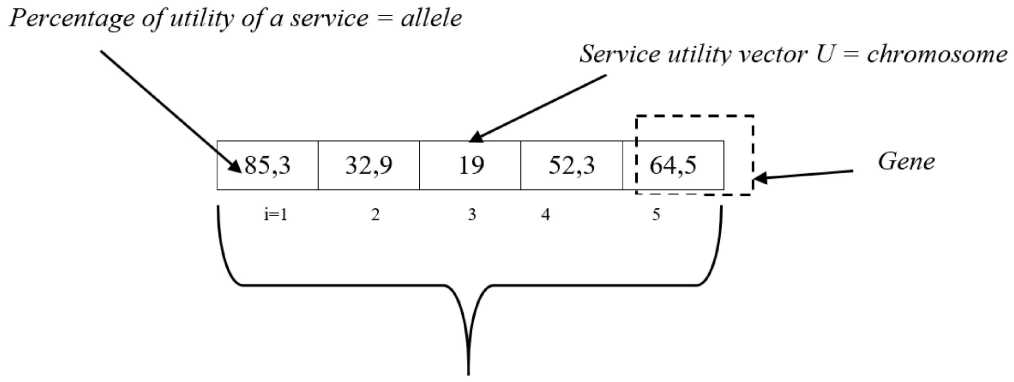


Figure 2: A chromosome.

ing rules of solutions, that is, the service utility vector is chosen randomly for each inter-organizational business process instance. A service utility vector is made up of a set of decision variables. These decision variables are the web services implementing the tasks of an inter-organizational business process. So, for each instance of an inter-organizational business process execution corresponds a service utility vector at the end of the execution. After multiple business process executions, we have multiple business process instances, which also correspond to multiple service utility vectors. In the case of the initial population, these service utility vectors are derived from service interaction matrices. This variability of the interaction matrix is due to the different screenshots made on the interacting information system at given instants $t_1, t_2, \ldots, t_n$; these screenshots represent an instant of execution of the business process studied. This means that at one instant t_1, we film or capture exchanges between the services of a business process under study, and then at another instant t_n, we capture yet more exchanges; we thus obtain different interaction matrices. In other words, at one instantt_1 of the interaction of interoperable information systems, we can obtain the interaction matrix MI_{t1}, and at another instant t_n, we can obtain another interaction matrix MI_{tn}. And the fitness value is defined by the interoperability cost optimization model like a penalized function

$$Fitness(U_{ti}) = f(U_{ti}) + R \times \sum \langle g_i(U_{ti}) \rangle, \tag{4.1}$$

R is the penalty factor and f is the objective function. By defining our fitness function using penalties, the idea of this method is to transform a constrained optimization problem into an unconstrained one by adding (or subtracting) a certain value to the objective function depending on the degree of constraint violation present in a certain solution. More information about penalty functions can be found in [6].

4.2 Genetic operators

4.2.1 Selection operator

Tournament selection is used to choose characteristics for the next generation. To apply this operator, the fitness value is calculated for each individual and the choice rate for each individual according to the formula

$$p(i) = \begin{cases} \binom{k}{n} \Big/ \binom{k}{n}, & \text{if} \quad i \in [1, n-k-1], \\ 0, & \text{if} \quad i \in [n-k, n] \end{cases} \tag{4.2}$$

with k the number of individuals selected from a large population of size n.

4.2.2 Mutation operator

We use the polynomial mutation operator. The authors [5] proposed a polynomial mutation operator with a user-defined indexing parameter (ηm). Based on a theoretical study, they concluded that (ηm) induces a perturbation effect of $\mathcal{O}((b-a)/(\eta m))$

in a variable, where a and b are the lower and upper bounds of the variable. They also found that a value $\eta m \in [20, 100]$ is adequate in most optimization problems. In this operator, a polynomial probability distribution is used to perturb a solution in the neighborhood of a parent. The probability distribution to the left and right of a variable value is adjusted so that no value outside the specified interval $[a, b]$ is created by the mutation operator. For a given parent solution $p \in [a, b]$, the mutated solution $p\prime$ for a particular variable for a random number u created in $[0, 1]$ is defined by [5]

$$p\prime = \begin{cases} p + \text{œ}_L(p - \mathrm{x}_i^{(L)}) & \text{for} \quad u \leq 0.5, \\ p + \text{œ}_R(\mathrm{x}_i^{(U)} - p) & \text{for} \quad u > 0.5. \end{cases} \tag{4.3}$$

4.2.3 Crossover operator

We use the simulated binary crossover (SBX) real crossover operator. The authors [4] was inspired by the single-point crossover operator applicable to two bit strings to develop this operator. Indeed, when in a genetic algorithm, we want to create child chromosomes from two binary parent strings $P1$ and $P2$ knowing that the crossover point lies between $[1, l - 1]$, with l the common length of the two bit strings, each of the child chromosomes $C1$ and $C2$ will have a sub-string from both parents. We can see that the child chromosomes lie inside or outside the region delimited by the parent points and the location of the crossover point. By finding the actual decoded value of the child chromosomes, we can see that there is a dispersion between values. Therefore, the authors define the spread factor as the ratio between the spread of the child points and that of the parent points as $\beta = |(C1 - C2)/(P1 - p2)|$ which is used to define the probability distribution as a function of β. More information about the SBX operator can be find in [4]

4.3 Elite-based learning mechanism and termination condition

The principle of elitism states that an individual's probability of reproduction is proportional to its relative fitness. Elitism refers to the fact that the best individuals must always participate in reproduction [10]. The learning strategy therefore enables the genetic algorithm to define a movement for exploring the best solutions in the search space. In our case, the learning strategy consists of reserving utility vectors for services where at least half of the genes have alleles with a percentage utility of the service greater than 50%. The termination condition of CostGA is defined when the number of generations is reached.

5 Procedure of the algorithm CostGA

Process of CostGA is described below.

(a) Initialization: generate an initial population whose size corresponds to the number of execution instances of the business process under study, given that θ is the number of services used by this process.

(b) Calculate the value of the fitness function for each service utility vector of the current generation and retain the best vectors.

(c) Select the parent vectors using the stochastic tournament selection operator.

(d) Modify the current generation by applying the SBX crossover operator with probability *pc* to the vectors.

(e) Modify the generation by applying the mutation operator with probability *pm* to the vectors.

(f) Use the learning mechanism to obtain the best utility vectors for the services of the current generation.

(g) Replace the vectors reserved in step (b.) with the best vectors obtained after the elitism mechanism to create a new generation.

(h) Count the number of generations *Ng*. If *Ng* reaches the predefined threshold Ngmax, the algorithm ends, otherwise, go to step (b).

This genetic algorithm enabled us to code an optimizer called CostGA in the C language. This optimizer optimizes the overall cost of interoperability for business processes whose tasks are implemented by web services with quality of service attributes such as response time and throughput.

6 Case study

The aim of this case study is to optimize the overall cost of interoperability in an enterprise network by analyzing the inter-organizational business process of cash withdrawal. The goal is to identify the set of unnecessary services that consume the most resources and obscure the execution of several process instances. To this end, we present the case study, define and apply the interoperability cost optimization approach, analyze the results obtained, and verify the convergence properties of the CostGA optimizer.

6.1 Presentation of the case study

The case study described here presents the collaboration and interoperability between the GIMAC (Interbank Electronic Payment Group of Central Africa) information system and the information systems of two partner banks, A and B. GIMAC is a banking ecosystem made up of a group of financial institutions, banks, mobile money operators, and aggregators present in all the countries of the CEMAC (Central African Economic and Monetary Community) sub-region. GIMAC's main mission is to implement full interoperability and interbanking for CEMAC electronic payment systems. With its GIMACPAY digital service, electronic financial transactions are federated (federated approach to interoperability), facilitating the transfer of money from one mobile account to another mobile account (or bank account) of another operator, and vice versa. In this study, we focus on the inter-organizational business process of

withdrawing money from GIMAC ATMs. This business process involves the interoperation of three different information systems: Bank A's information system, Bank B's information system, GIMAC's information system, and an *X* customer. Data collection was based on an interview with GIMAC's mobile financial services engineer concerning the GIMACPAY solution (which federates all electronic payment systems and means in CEMAC) and the inter-organizational business process studied. Information from the GIMAC website was also used to complete our analysis. During a period of immersion within the company, we worked with the expert to learn more about the business process under study.

(a) Relationships between the companies studied Customer X needs to withdraw money from his bank account A (Bank A's information system) at Bank B's ATM (Bank B's information system) using his GIMACPAY card (GIMAC's information system). Although all the contextual elements (e.g. the amount to be withdrawn is less than the account amount) are correct, the transaction failed. For this reason, we are interested in studying the inter-organizational business process of cash withdrawal. To find out more, let us define the process scenario.

(b) Process scenario. The process of withdrawing cash from a GIMAC ATM consists of two sub-processes: the process of obtaining cash and the process of monitoring the transaction. The process of obtaining cash starts when customer X inserts his card into an authorized Bank B ATM. The ATM checks the card's compliance with the GIMAC system, and also verifies the customer's balance. If the card complies and the required amount is less than the current balance, the ATM debits the account. It is then up to the customer to retrieve their card and take their cash. If this is not the case, no transaction is carried out, and the customer simply gets his card back. In either case, the transaction is recorded in the interoperable information system. The transaction tracking process is as follows: based on the transaction carried out by the customer, a transaction report is drawn up for each customer making a withdrawal, according to date (time, month, year), bank (issuing bank and receiving bank) and location (country, city) where the transactions were carried out. When a withdrawal fails, the bank information system associated with the ATM receives notification of the failed transaction; it checks and identifies the origin of the card, then transmits a report (an automatic transaction) to the GIMAC information system, which notifies the bank owning the customer account.

6.2 Definition and application of the global interoperability cost optimization approach

The approach to optimizing the overall cost of interoperability, based on a business process analysis, consists of four stages:

(a) Model the inter-organizational business process under study.

(b) Identify the optimizer's inputs: The services and number of services executing the tasks of an instance of the business process under study; the response time (mini-

mum and maximum) and throughput (minimum and maximum) intervals relative to the execution of all the process tasks.

(c) Apply the CostGA optimizer to obtain the overall interoperability cost of the business process and the best solution vector or service utility vector.

(d) Interpretation of results: Browse the components of the solution vector to identify useful web services, and interpret the resulting global cost of business process interoperability.

The Table 1 gives the correspondences between the concepts of genetic algorithms and the personalization of these concepts in our work.

Table 1: Correspondence between genetic algorithm concepts and customized concepts.

Genetic algorithm concepts	Customization of concepts in our work
Population size	Number of times a business process is executed (number of instances)
Objective function	Interoperability cost optimization model
Chromosome	Service utility vector
Gene position	Service identifier
Population	Set of service utility vectors
Allele	Percentage of service utility
Number of genes	Number of services executing the tasks of an inter-organizational business process
Number of Generations	Program stop condition

The business process interoperability cost optimization approach helps us to make better use of the defined optimizer in a real environment, and even better within the network of enterprises that is the subject of our study. It serves as a transition between the defined theoretical model and its practical application. In the following sections, we apply this approach to analyze the overall cost of interoperability in the cash withdrawal process under study.

6.2.1 Model the inter-organizational business process under study

We have used BPNM notation to model this process. Knowing that we are interested in the interoperability of business processes, i.e. in the analysis of interoperable information systems in execution, and that we cannot access the real GIMAC execution environment due to the sensitivity of the financial data contained in GIMAC, we proceeded to simulate the cash withdrawal process using a business process simulator called "Bizagi Modeler" in a scenario close to the reality of the interoperable information system analyzed. The aim of this simulation was to gain insight into the minimum and maximum response times, and the minimum and maximum throughput associated with each service executing a task of the business process instances studied. To this end, one hundred instances of the business process under study were simulated in order to identify the maximum and minimum response times, bearing in mind that

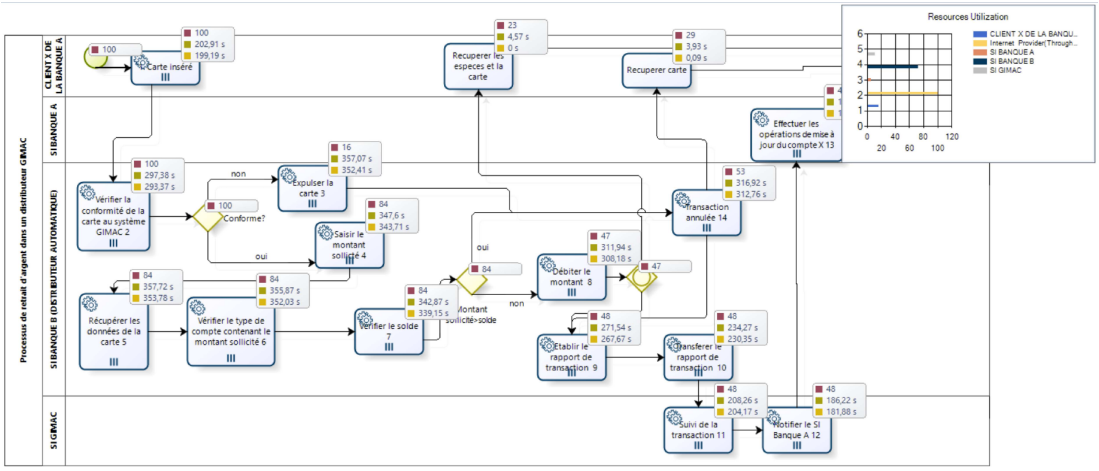


Figure 3: Simulation of the cash withdrawal process using different resources.

the minimum and maximum throughput was considered as a resource provided by an Internet service provider. Fig. 3 shows the process studied and the various simulations linked to resource consumption in this process.After simulating 100 instances of the cash withdrawal process, we deduce that the minimum response time is 11 min 54 s and the maximum response time is 54 min 58 s. The minimum throughput is 2 Mbps and the maximum is 622 Mbps.

6.2.2 Identify the optimizer's inputs

The Table 2 contains the inputs values of the optimizer.

Table 2: Identify the optimizer's inputs.

Parameters	Values
Population size	100
Number of generations	150
Number of real variables or numbers of services	14
Probability of crossover Pc	0.9
Probability of mutation Pm	0.5
Exponent for polynomial crossover operator n	2
Exponent for polynomial mutation operator ηm	100
Static penalty factor R	10

6.2.3 Apply the CostGA optimizer to obtain the overall interoperability cost of the business process and the best solution vector or service utility vector

The characteristics of the machine we used are as follows: Intel(R) Core(TM) i5-8265U CPU @ 1.80 GHz, C language and GCC compiler (MinGW.org GCC-6.3.0-1) 6.3.0 to develop the CostGA optimizer and perform the tests related to our case study. Thus,

after five experiments, we notice that the fourth experiment (see Fig. 4) produces better results. These results are shown in Table 3. Analysis of these results reveals that services 4, 10 and 11 are not useful for the interoperability of information systems with the cash withdrawal process studied, and the overall interoperability cost of this business process is −125.934474.

```
Run No. 4
==================================================
|
Best ever fitness: -125.934474 (from generation : 2)
Variable vector: Binary | Real -> | 0.844120 0.920952 0.968577 0.413687 0.836393 0.618520 0.912130 0.998228 0.991515 0.327924 0.317986 0.892030 1.000000 1.000000
Constraint value:| Overall penalty: 0.000000
==================================================
```

Figure 4: Results of the fourth experiment.

Table 3: Results of interoperability cost optimization of the information systems of the GIMAC business network through the analysis of the cash withdrawal business process in a GIMAC ATM.

Service id	Service percentage utility
1	0.844120
2	0.920952
3	0.968577
4	0.413687
5	0.836393
6	0.618520
7	0.912130
8	0.998228
9	0.991515
10	0.327924
11	0.317986
12	0.892030
13	1.000000
14	1.000000

6.2.4 Interpretation of results and analysis of the optimizer's convergence property

We have used the genetic algorithm so that, we refer to the field of biology to understand how the best fitness value can be interpreted, knowing that this value represents the overall cost of interoperability in the context of our work. Biology field distinguishes four types of fitness value: tautological fitness, Darwinian fitness, Thoday fitness and inclusive fitness. In each of these categories, the definition of fitness divides scientific opinion. However, in our work, we take into consideration the definitions of

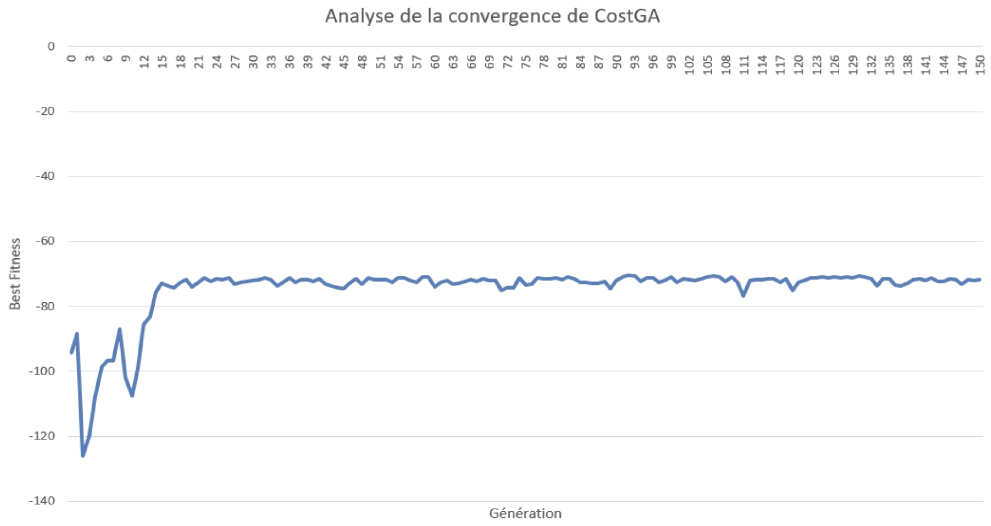


Figure 5: CostGA optimizer convergence analysis.

Darwin's and Thoday's work. For Darwin, fitness is the expectation of offspring, or rather an expectation of a system, and for Thoday, fitness is a unit of evolution [9]. On the basis of these ideas, we note that fitness value is not a probability and has no unit of measurement, which supports the results we have obtained. Fitness value must therefore be understood as being relative to a particular environment at a given point in time. The fitness value is only defined over one generation. Thus, a fitness function that takes an input X, where X is a vector with as many elements as the number of variables in the problem, calculates the function's value and returns this scalar value in its return argument Y. The fitness value of an individual is the value of the fitness function for that individual. Thus, the best fitness value is the smallest fitness value for any individual in the population. As we have already pointed out, the best fitness value corresponds to the overall cost of interoperability. In the context of our work, this value is equivalent to a mathematical definition of the fitness value. Thus, within the framework of mathematical definitions of the fitness value, this value can take on a series of different values. Sometimes, fitness values extend over positive real numbers, sometimes over real numbers between 0 and 1, sometimes over real numbers greater than or equal to 1, and this fitness value is a dimensionless quantity [9], i.e. it has no unit of measurement. Thus, in our work, we consider the case where the fitness value is greater than or equal to 1 to better interpret the overall cost of interoperability, and we define the following interpretation rules:

- If Best fitness = global cost of interoperability is less than one, then the business process is not adapted to the information system environment and does not allow the interoperable information system to evolve.
- If Best fitness = global cost of interoperability is greater than or equal to one, then the business process is adapted to the information system environment and supports the evolution of the interoperable information system.

In the case of the inter-organizational cash withdrawal business process studied, given that the overall cost of business process interoperability is $-125.934474 < 1$, we conclude that this process is not adapted to the environment of the interoperable information system. According to Fig. 5, which shows that this value is the best solution found by the optimizer after eighteen generations, for the analysis of the global cost of interoperability of the business process studied. The complexity of the algorithm is of the order of $\mathcal{O}(gnm)$ with g the number of generations, n the population size and m the individual size. We have $g = 150$, $n = 100$ and $m = 14$.

7 Conclusion

In this article, the problem we address is to show how to optimize interoperability cost within a networked enterprise. To solve this problem, we have generated new knowledge in our field of information systems by developing an interoperability cost optimization model, defining and implementing in C a global interoperability cost optimizer for business processes using the genetic algorithm. These contributions were validated in practice within a network of companies called GIMAC, through the study of the inter-organizational business process of cash withdrawal in a GIMAC-approved distributor. The results show that the overall cost of interoperability is negative, which proves that the interoperable information system of this business network will no longer be scalable after 18 generations. As a continuation of this work, we plan to reconsider certain parts of the program (the genetic algorithm) running in the optimizer, in order to apply parallelism to the execution of in-memory tasks; to test the optimizer by implementing other crossover and mutation operators; and to reconsider other quality-of-service attributes such as reliability, in order to calculate the interoperability cost of a single service.

References

[1] J. O. Agushaka et al., *Effect of weighting scheme to QoS properties in Web service discovery*, International Journal of Computer Science and Information Security, 2010.
[2] C. Berg, *Interoperability*, Internet Policy Review, 13(2), 2024. doi: 10.14763/2024.2.1749.
[3] N. Daclin, D. Chen, and B. Vallespir, *Developing enterprise collaboration: A methodology to implement and improve interoperability*, Enterp. Inf. Syst., 10 (2014), pp. 467–504.
[4] K. Deb and R. Bhushan Agrawal, *Simulated binary crossover for continuous search space*, Complex Syst., 9(2) (1995), pp. 115–148.
[5] K. Deb and D. Deb, *Analysing mutation schemes for real-parameter genetic algorithms*, Int. J. Artif. Intell. Soft Comput., 4(1) (2014), pp. 1–28.
[6] A. E. Eiben and J. E. Smith, *Introduction to Evolutionary Computing*, in: Natural Computing Series, Springer, 2015.
[7] S. Katoch, S. S. Chauhan, and V. Kumar, *A review on genetic algorithm: Past, present, and future*, Multimed. Tools Appl., 80 (2021), pp. 8091–8126.

[8] B. Kazimipour, X. Li, and A. K. Qin, *A review of population initialization techniques for evolutionary algorithms*, in: 2014 IEEE Congress on Evolutionary Computation (CEC), (2014), pp. 2585–2592.
[9] S. Kimbrough, *The concepts of fitness and selection in evolutionary biology*, J. Social Biol. Struct., 3 (1980), pp. 149–170.
[10] N. Saini, *Review of selection methods in genetic algorithms*, Int. J. Eng. Comput. Sci., 6(12) (2017), pp. 22261–22263.
[11] A. H. Wright, *Genetic algorithms for real parameter optimization*, in: Foundations of Genetic Algorithms, Elsevier, (1991), pp. 205–218.