9th International Food Operations & Processing Simulation Workshop
20th International Multidisciplinary Modeling & Simulation Multiconference

2724-0355 © 2023 The Authors.
doi: 10.46354/i3m.2023.foodops.008

An asexual genetic algorithm for the smallholders' demand selection problem

Manuella Germanos[1,*], Oussama Ben-Ammar[2] and Gregory Zacharewicz[1]

[1]Laboratory for the Science of Risks, IMT Mines Alès, France
[2]EuroMov Digital Health in Motion, Univ Montpellier, IMT Mines Alès, Alès, France

[*]Corresponding author. Email address: manuella.germanos@mines-ales.fr

Abstract

As food safety shakes due to climate change and the looming possibility of a calamity similar to that of the COVID-19 pandemic, along with the increase in the world's population, there is an immediate need to increase food production by optimizing the agri-food supply chain. This work aims to help small-scale farmers manage this increase in orders and help them gear their resources towards more profitable practices by employing a multi-objective optimization model based on genetic algorithms that considers environmental aspects. To do so, we implement an asexual genetic algorithm that takes as input the demands received by the farmer and outputs the best combinations of demands to meet. The model takes into consideration the amount of land available for cultivation, as well as the resources (water, cost of cultivation, etc.) and revenue of the demands to determine the best combinations of demands to meet. This work is developed in the SMALLDERS framework and builds over the scarce literature that tackled the demand selection problem in the agricultural field.

Keywords: Genetic Algorithm; Multi-objective optimization; Agri-food supply chain; demand selection.

1. Introduction and problem description

As the COVID-19 pandemic recedes, its impact on the economy persists. This pandemic revealed how fragile the global and local supply chains are (Aday and Aday, 2020). With the increase in world population and the risk of climate change looming near, there is a dire need to optimize the agri-food supply chain to ensure food safety. This necessity helped drive the (SMALLDERS) project, a European project that aims to assist small-scale farmers and increase their resilience against upcoming challenges. The work presented here comes as a part of the SMALLDERS project and aims to help small-scale farmers, or what we will refer to as smallholders, better manage the demands they receive.

There has been a recent increase in the demand for smallholder products. This comes in part from the increase in demand in general, but it is also due to the shift in interest from mass-produced sustenance to more locally produced ones. With this shift in demand, some smallholders find themselves unable to meet all the clients' needs and are obligated to choose which demands to satisfy. In this work, we tackle the demand selection problem for small-scale farmers. We attempt to maximize the farmer's revenue by helping them choose the right products to plant and minimize the storage cost. Furthermore, we take into consideration the amount of workforce needed to complete a demand as some smallholders are facing a shortage of workers. Finally, we focus on minimizing water use to assist the smallholders working in drought-prone areas. To do so, we propose an asexual genetic algorithm that helps guide the smallholder toward the products to cultivate.

This work is not the first to tackle the demand selection problem. We reference some of the research work that tackled this problem. The work of Shu et al. (2013) studied which demand a manufacturer should satisfy given a single-item, multi-echelon inventory distribution system. The authors developed a decision-making system that uses mathematical models and takes into consideration the cost and revenue of the items to choose which demand to satisfy. In Geunes and Geunes (2012), the authors developed a decision-making model that helped determine which demand to meet and better manage the inventory of a single-item firm. Their approach employed a linear programming model and a heuristic to maximise profit. In Silva et al. (2018), the authors tackled the order of acceptance and scheduling problem by building a decision model to help choose which demands and orders to accept and the order by which they are produced. The work dealt with a single machine and multiple items and used an exact mathematical model that employs column generation and Lagrangian relaxation. In Leng et al. (2021), the authors employed deep learning and reinforcement learning to assist printed circuit board manufacturers in minimising their energy consumption and maximising their profit. The deep learning model helped predict incoming orders' production cost, makespan, and carbon consumption, whereas the reinforcement learning model decided which orders to accept. In Dumetz et al. (2017), the authors focused on manufacturers that produce multiple products from the same raw material. They used a simulation model that integrates custom-built enterprise resource planning to evaluate different approaches for the North American lumber industry. Their work showed that different scenarios require different order acceptance policies. The work in Li et al. (2019) focused on the order acceptance and scheduling problem for the additive manufacturing industry. The authors proposed a heuristic to tackle this problem and maximise the average profit-per-unit-time and combined global and local decision-making strategies. In Ebben et al. (2005), the authors worked on assessing the effectiveness of different methods used in order acceptance and production planning problems. The work considered production capacity, stochastic processing times, precedence relations, technological restrictions, and release and due dates of orders. The authors use a simulation model to assess various approaches and found that EDD-based order acceptance is usually preferred when looking at performance. Finally, in Aouam et al. (2018), the authors study the effects of different demands on the production of a multi-product firm. They argue that high-setup orders that cannot be aggregated with other orders and orders that increase the workload and cause delays in other orders should not be accepted. They develop a decision-making model that uses mixed integer programming, a relax and fix heuristic, and a fix and optimise heuristic to tackle the problem of demand selection.

The proposed method in this paper will also attempt to solve the demand selection and planning problem. The work will focus on the agricultural field while taking into consideration that a farmer can cultivate multiple products, whereby each product has different characteristics (unit price, cultivation time, hours of labour, and cost of storage). This work also takes into consideration environmental aspects and tries to minimise water usage. As seen previously, there is an abundance of research that tackled the demand selection problem. However, to the best of our knowledge, no other works tackled this problem for the short-food supply chain and smallholders.

The remainder of the paper is organised as follows. Section 2 introduces the concept of the Pareto front and the NSGA-II, which are proposed to optimise the demand selection. Section 3 introduces a variant of NSGA-II developed based on the asexuality concept. Section 4 gives a numerical example. Finally, Section 5 rounds off with a conclusion and perspectives for future research.

2. Optimization approach

In this study, every farmer will receive a list of demands six months in advance. The farmer will need to choose which products they are willing to farm. In this work, we employ a genetic algorithm to tackle this decision-making problem referred to as the demand selection problem (Geunes et al., 2005; Mohammadivojdan and Geunes, 2018). Although this model is developed to assist farmers in demand selection and to plan their farming schedule, this approach can be generalised to any field that requires demand selection and a division of the work resources.

In the following subsection, we introduce the Pareto front that we use to optimise the problem using the non-dominated Sorting Genetic Algorithm II.

2.1. Pareto front

The Pareto front is the set of non-dominated, feasible solutions of a given search space. A solution is referred to as non-dominated if we cannot improve any performance indicator without worsening the performance on another one (Akbari et al., 2014).

For example, given the Pareto front of Figure 1 that shows the performance of different solutions on Indicators 1 and 2. The objective is to minimize these indicators. The solutions shown in red are the non-dominated solutions as no other solution performs better on one indicator without sacrificing the performance on the other. The blue data points do not belong to the Pareto front as other solutions dominate them, i.e., there exists another solution that outperforms it on one performance indicator and performs just as well, or even better, on the other one.

In this work, we need an optimisation approach to rank the solutions on different indicators based on the Pareto front. To do this, we developed a non-dominated Sorting Genetic Algorithm II that is detailed in the following subsection.

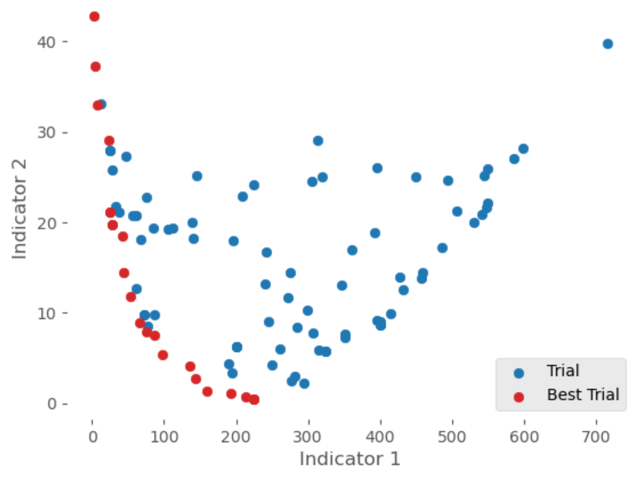

Figure 1. Pareto front example.

2.2. Non-dominated Sorting Genetic Algorithm II

The non-dominated sorting genetic algorithm II (NSGA-II) is a famous variant of the genetic algorithm implemented when dealing with multi-objective optimisation Deb et al. (2002). The chromosomes in NSGA-II are clustered depending on the number of solutions that can dominate them. The chromosomes that belong to the Pareto front are labelled as belonging to front F_1. Lesser performing chromosomes are clustered into fronts $F_2, F_3, \cdots, F_n$, depending on the number of solutions that can dominate them.

In the following section, we introduce the asexuality concept, which is believed to help the heuristic reach good solutions faster (Salesi et al., 2021).

3. Asexual variant of NSGA-II

Our proposed model uses the asexuality concept of genetic algorithms whereby one chromosome goes through mutation operators to generate one offspring. This differs from its classical counterpart, which combines crossover operators and mutation operators to create the progeny. The reasoning behind this choice is that the crossover operator is typically heavy computationally, and using only the mutation algorithm might result in a good progeny at a faster pace Cantó et al. (2009).

The chromosomes in this approach are a list of six genes. Each gene maps back to a month and contains the orders that are processed (the plants cultivated) during that month. If a product's cultivation period spans more than one month, the product will be found in consecutive months in the chromosome. For example, let's assume that the farmer will cultivate the produce of Table 1. One chromosome that represents a possible solution is shown in Table 2. In this example, we cannot cultivate Product A and Product B simultaneously as it would require more land than what is available.

Table 1. List of possible demands

Product	Time needed	% Land allocation
Product A	2 months	60%
Product B	4 months	50%
Product C	3 months	20%

Table 2. Chromosome representation of a possible solution to demand set in Table 1

*M*0	*M*1	*M*2	*M*3	*M*4	*M*5
B	B	B	B C	A C	A C

3.1. Initialization, fitness, and selection

To initialise the population, we start with randomly generated chromosomes that respect the land size constraint, whereby they don't over-allocate land.

Each chromosome's fitness depends on the expected revenue from the completed demands, the amount of work hours needed to cultivate the demand, the cost of storing the produce till its delivery after month 6, and the amount of water used to develop them. The objective function aims to maximise profit while minimising the work hours and the amount of water used. To assess the performance of a given solution, it is compared to all the other solutions of the given population or solution space. The solutions found to be non-dominated are labelled as the Pareto front of the space or the F_1 group. The solutions dominated by a single solution are labelled into group F_2. So on and so forth until all the solutions are labelled into their respective groups (Deb et al., 2002).

To generate the next generation of chromosomes, called progeny, we select chromosomes using the tournament selection method (Deb et al., 2002).

3.2. Mutation operator

Next, we perform mutations to create new solutions. To do so, we propose the following simple mutations. Given a chromosome solution, perform a combination of the following operations:

- **Mutation 1:** Remove the product that requires the most cultivation time.
- **Mutation 2:** Remove the most expensive product to cultivate.
- **Mutation 3:** Remove the product that required the most land space to cultivate.
- **Mutation 4:** Remove a random product.
- **Mutation 5:** Move the product that is the most expensive to store towards the end of the month and rearrange the demand using the "As-Late-As-Possible" algorithm.
- **Mutation 6:** Randomly add demands to the chromosome that does not break the land allocation constraint.

These proposed operations were chosen to balance adding and removing demands from the solution and moving the cultivation schedule across the months. The genetic algorithm randomly chooses which mutation operations should be performed on a given chromosome and in which order. Once the offspring is generated, its fitness is assessed, and it is added to the population.

3.3. Natural Selection, perturbation, and stopping criteria

As more progeny is added to the population, its size will grow. Once the population has doubled in size, we eliminate the worst half of the population and keep the better-performing half. To do so, we first rank the chromosomes by the front they belong to and then rank the chromosomes within the same front by their crowding distance Deb et al. (2002)—the crowding distance measures how close a solution is to its neighbours in the front. Chromosomes with larger crowding distances are preferred since they map to distant points in the search space. Then, the population is shrunk to its original size by keeping the best fronts first, and then the chromosomes with the highest crowing distance in the front in case the entire front cannot fit in the shrunken population.

The process of growing the population and shrinking it in size is repeated multiple times to create multiple generations of solutions and allow the exploitation and exploration of the search space.

However, genetic algorithms are prone to stagnation Fogel (1994). To avoid this, we introduce perturbation to the population whereby the elite chromosomes, or in this case, the chromosomes of the F1 front, are preserved, but the remaining population is discarded. The population is then grown back to its original size by adding randomly generated chromosomes. This procedure is employed with a low probability after every generation (Moscato and Cotta, 2010). Figure 2 illustrates this process. Starting from the initial population, a set of progeny is created. This doubles the size of the population. Then, the solutions are sorted into fronts to shrink the population size. In this example, fronts F_1 and F_2 can fit entirely in the new population. However, not all solutions in front F_3 can make the cut. Therefore, the model takes the top solutions with the biggest crowding distance; the remaining solutions in F_3 and all the solutions in F_4 are discarded. This results in a new population that contains better-performing solutions.

This process is repeated until a set number of generations are grown.

4. Numerical Example

As an example of an application for the model, we run the asexual genetic algorithm on the problem instance described in Table 3. This table shows a list of demands with the time needed to cultivate the products (in months), the percent of land that needs to be allocated to meet this demand, the amount of water needed for the demand (in units of 100 Litres), the total hours of work required to cultivate the demand, the revenue of the farmer from this demand (in units of 1000 €), and the storage costs of the demand (in units of 1000 €) respectively.

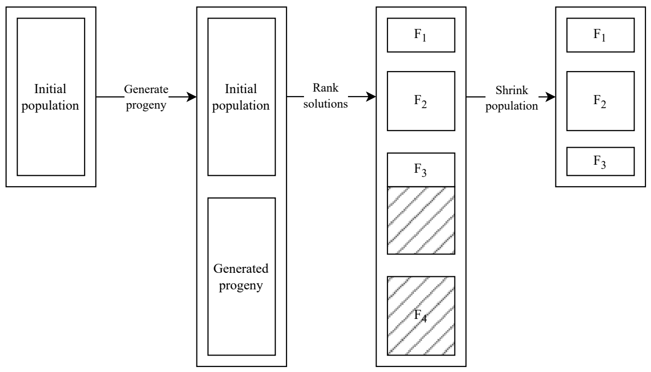

Figure 2. Growing and shrinking the population

Table 3. List of demands for a problem instance

Time	% Land	Water	Hrs of Work	Revenue	Storage cost
3	30	350	30	10	2
3	20	220	25	15	3
4	25	300	40	20	2
4	20	550	42	25	2
3	30	420	28	10	6
3	10	230	31	50	1
2	15	370	20	15	2
4	7	580	44	10	2
2	15	370	27	10	2
5	25	110	51	13	5

Table 4. List of solutions found by the model

	Water	Hrs of Work	Profit
Sol 1	**2430**	240	134
Sol 2	2740	**217**	131
Sol 3	3200	298	**146**

The proposed model was implemented using Python language. The test was carried out with a 12th Gen Intel(R) Core(TM) i9-12900H processor and 64 GB of RAM. Once executed, the model returns the solutions that belong to the Pareto front. As an example of some solutions found, we show in Table 4 the chromosomes that performed best for every indicator and highlight this performance in bold.

After returning the indicators of every solution, along with its cultivation plan, the farmer would then choose the solution that best fits their objectives and constraints. For example, if a farmer has water resource limitations but can easily find the workforce needed to cultivate the plants, they might gravitate towards the first solution in the table, which minimises the amount of water necessary to meet the demand and maximises the revenue, without worrying labour shortage

5. Conclusion and Future work

In this work, we tackle the demand selection problem for small-scale farmers using genetic algorithms. Our work focuses on maximising profit while minimising the cost of cultivation, the hours of work, and the amount of water used to cultivate the produce. Since the objective function of this work contains multiple indicators, we implement the NSGA-II model to rank the solutions into different fronts and select our optimal solutions as belonging to the Pareto front. Our approach also uses the asexuality variation whereby mutation operators are employed on a chromosome to generate its progeny.

One limitation of this model is that it overlooks weather factors that impact the farmer's production. For example, this model does not consider that some products must be farmed during specific periods. The model does not take into consideration possible disruptions and crises that the farmer might face, such as periods of drought.

A future direction this work can take is monitoring the conditions of the land using sensors or weather forecasts. The model is also limited as it considers only one farmer. An expansion of the model would be to take into consideration the community of farmers and divide the demands among them so that there is an equal division of profit among them and no overproduction of certain items.

6. Acknowledgments

This work is part of the research project entitled "Smart Models for Agrifood Local vaLue chain based on Digital technologies for Enabling covid-19 Resilience and Sustainability" (SMALLDERS), co-funded by the PRIMA Program - Section 2 Call multi-topics 2021, through the following National Authorities: Ministry of Universities and Research (MUR, Italy), State Research Agency (AEI, Spain), Agence Nationale de la Recherche (ANR, France), Ministry of Higher Education and Scientific Research (Tunisia).

The authors acknowledge the grant *PCI2022132924* funded by MCIN/AEI/ 10.13039/501100011033 and by the "European Union NextGenerationEU/PRTR".

References

Aday, S. and Aday, M. S. (2020). Impact of covid-19 on the food supply chain. *Food Quality and Safety*, 4(4):167–180.

Akbari, M., Asadi, P., Besharati Givi, M., and Khodabandehlouie, G. (2014). Artificial neural network and optimization. *Advances in friction-stir welding and processing*, pages 543–599.

Aouam, T., Geryl, K., Kumar, K., and Brahimi, N. (2018). Production planning with order acceptance and demand uncertainty. *Computers & Operations Research*, 91:145–159.

Cantó, J., Curiel, S., and Martínez-Gómez, E. (2009). A simple algorithm for optimization and model fitting: Aga (asexual genetic algorithm). *Astronomy & Astrophysics*, 501(3):1259–1268.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197.

Dumetz, L., Gaudreault, J., Thomas, A., Lehoux, N., Marier, P., and El-Haouzi, H. (2017). Evaluating order acceptance policies for divergent production systems with co-production. *International Journal of Production Research*, 55(13):3631–3643.

Ebben, M. J., Hans, E. W., and Olde Weghuis, F. (2005). Workload based order acceptance in job shop environments. *OR spectrum*, 27:107–122.

Fogel, D. (1994). Asymptotic convergence properties of genetic algorithms and evolutionary programming: Analysis and experiments. *Cybernetics and Systems*, 25:389–407.

Geunes, J. and Geunes, J. (2012). Dynamic lot sizing with demand selection and the pricing analog. *Demand Flexibility in Supply Chain Planning*, pages 41–50.

Geunes, J., Merzifonluoğlu, Y., Romeijn, H. E., and Taaffe, K. (2005). Demand selection and assignment problems in supply chain planning. In *Emerging Theory, Methods, and Applications*, pages 124–141. INFORMS.

Leng, J., Ruan, G., Song, Y., Liu, Q., Fu, Y., Ding, K., and Chen, X. (2021). A loosely-coupled deep reinforcement learning approach for order acceptance decision of mass-individualized printed circuit board manufacturing in industry 4.0. *Journal of cleaner production*, 280:124405.

Li, Q., Zhang, D., Wang, S., and Kucukkoc, I. (2019). A dynamic order acceptance and scheduling approach for additive manufacturing on-demand production. *The International Journal of Advanced Manufacturing Technology*, 105:3711–3729.

Mohammadivojdan, R. and Geunes, J. (2018). Supply and demand selection problems in supply chain planning. *Open Problems in Optimization and Data Analysis*, pages 61–82.

Moscato, P. and Cotta, C. (2010). A modern introduction to memetic algorithms. *Handbook of metaheuristics*, pages 141–183.

Salesi, S., Cosma, G., and Mavrovouniotis, M. (2021). Taga: Tabu asexual genetic algorithm embedded in a filter/filter feature selection approach for high-dimensional data. *Information Sciences*, 565:105–127.

Shu, J., Li, Z., and Huang, L. (2013). Demand selection decisions for a multi-echelon inventory distribution system. *Journal of the Operational Research Society*, 64:1307–1313.

Silva, Y. L. T., Subramanian, A., and Pessoa, A. A. (2018). Exact and heuristic algorithms for order acceptance and scheduling with sequence-dependent setup times. *Computers & operations research*, 90:142–160.