AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE
Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej

Projekt dyplomowy

Modelowanie i optymalizacja produkcji rolnej
Modeling and Optimization of Agricultural Production

Autor: Piotr Paweł Hudaszek
Kierunek studiów: Automatyka i Robotyka
Opiekun pracy: Dr inż. Piotr Kadłuczka

Kraków, 2022

Spis treści

Wstęp

<TODO>

Program ma za zadanie zoptymalizować terminarz upraw na określonej liczbie pól tak aby jak najlepiej wykorzystać dostępne zasoby. Został on napisany w Pythonie jako że jest to jeden z bardziej popularnych języków programowania który wciąż się rozwija i zyskuje popularność [PYPL]. Jako algorytm optymalizacji wybrano algorytm genetyczny.

W pierwszym rozdziale został opisany model matematyczny wraz z rozważaniem na temat użytych uproszczeń i przydatności modelu w realnych zastosowaniach. Rozdział drugi jest poświęcony przystosowaniu algorytmu genetycznego do podanego problemu w tym opisanie użytych operatorów genetycznych. W dalszej części pracy zostały zaprezentowane rozwiązania implementacyjne oraz przeprowadzone testy.

1 Opis zagadnienia

1.1 Model opisowy

Opisywane zagadnienie polega na optymalizacji zysku z uprawy w przedsiębiorstwie rolnym. Jest możliwość uprawy różnych rodzajów produktów. Każdy rodzaj potrzebuje określone zasoby na danym etapie uprawy. Przedsiębiorca ma dostęp do pewnej ilości każdego rodzaju zasobów. Zasoby dzielą się na dwie klasy: zasoby dzienne które są przydzielone na dany dzień oraz zasoby całkowite przydzielone na cały rozpatrywany okres uprawy. Jeśli uprawa zostanie przeprowadzona i zasobów nie zabraknie to przedsiębiorca uzyska określony bazowy zysk z jednostki pola, który zależny jest od rodzaju produktu. Rozpatrywane przedsiębiorstwo ma określoną ilość pól. Każde pole można opisać jego obszarem oraz współczynnikami które określają jak dobrze dane pole pasuje do uprawy danego rodzaju produktu. Aby uzyskać rzeczywisty zysk należy przemnożyć zysk bazowy z współczynnikiem dopasowania oraz wielkością pola.

Optymalizacja polega na wyborze jaki produkt jest uprawiany na danym polu oraz kiedy rozstanie rozpoczęta uprawa. Na polu można uprawiać kilka produktów po sobie. Data rozpoczęcia uprawy jest ograniczona przez ramy czasowe danego produktu.

1.2 Model matematyczny

Dane:

$t \in T$ - dzień

$p \in P$ - pole dostępne do uprawy

S_p - powierzchnia pola p

y_{pn} - rodzaj n-tej rozpoczętej uprawy na polu p

x_{pn} - dzień rozpoczęcia n-tej uprawy na polu p

d_y - dochód z jednostki pola na którym był uprawiany produkt y

w_{py} - współczynnik dopasowania pola p do produktu y

$z_d \in Z_d$ - dostępny zasób dzienny (odnawialny po każdym dniu)

$z_c \in Z_c$ - dostępny zasób całkowity (przydzielony na cały rozpatrywany okres uprawy)

z_{dyt} - zasoby dzienne potrzebne do uprawy produktu y w t dniu uprawy

z_{cyt} - zasoby całkowite potrzebne do uprawy produktu y w t dniu uprawy

t_{y0} - pierwszy dzień w którym można zacząć uprawę produktu y

t_{yN} - ostatni dzień w którym można zacząć uprawę produktu y

o_y - czas trwania uprawy produktu y

Postać rozwiązania:

Lista dwu elementowych wektorów dla każdego pola gdzie pierwszy element x oznacza dzień rozpoczęcia a drugi y rodzaj uprawy.

$$\begin{Bmatrix} (x_{11}, y_{11}) & (x_{12}, y_{12}) & \cdots & (x_{1N_1}, y_{1N_1}) \\ (x_{21}, y_{21}) & (x_{22}, y_{22}) & \cdots & (x_{2N_2}, y_{2N_2}) \\ \vdots & \vdots & \ddots & \vdots \\ (x_{P1}, y_{P1}) & (x_{P2}, y_{P2}) & \cdots & (x_{PN_P}, y_{PN_P}) \end{Bmatrix}$$

Funkcja celu:

Zysk ze sprzedaży uzyskanych produktów .

$$f(x_{pn}, y_{pn}) = \sum_{p \in P} \sum_{n \in N_p} S_p d_{y_{pn}} w_{py_{pn}} \rightarrow max$$

Ograniczenia:

Dla każdego dnia t każdego rodzaju zasobu dziennego ilość użyta nie może przekroczyć dostępnej.

$$\forall z_d \in Z_d \; \forall t \in T \quad \sum_{p \in P} z_{dtp} \leq z_d$$

Dla każdego rodzaju zasobu całkowitego ilość użyta przez cały okres uprawy nie może przekroczyć dostępnej.

$$\forall z_c \in Z_c \quad \sum_{t \in T} \sum_{p \in P} z_{ctp} \leq z_c$$

Uprawę produktu typu y można zacząć w wyznaczonym oknie czasowym.

$$t_{y0} \leq o_y \leq t_{yN}$$

W danym dniu na jednym polu można uprawiać tylko jeden rodzaj produktu.

$$\forall p \in P \; \forall n \in N_p \quad x_{pn} + d_y < x_{pn+1}$$

1.3 Model a rzeczywisty problem

Tak sformułowany model pozwala oddać typowe zależności podczas planowania uprawy tak jak ograniczenie terminu rozpoczęcia uprawy i konieczne zasoby. Ilość zasobów nie jest zdefiniowana na sztywno pozwala to użytkownikowi programu wprowadzić zasoby specyficzne dla danego typu uprawy i sprawić że program jest bardziej uniwersalny. Model można też zastosować do innych zagadnień niż rolnicze na przykład do bilansowania zasobów w zakładzie produkcyjnym.

Model nie bierze pod uwagę wpływu nieprzewidywalnych zmian cen oraz pogody. Natomiast użytkownik może wprowadzić odpowiednio większe potrzebne ilości zasobów lub dłuższy czas uprawy aby mieć pewien margines bezpieczeństwa.

Mimo tych ograniczeń uważam że model może być przydatny. Szczególnie w uprawie szklarniowej gdzie wpływ pogody jest ograniczony oraz w przypadku gdy optymalne wykorzystanie któregoś z zasobów jest kluczowe. Tak jest na przykład w przypadku rejonów z ograniczoną ilością wody[Optimal Use of Irrigation Water].

2 Opracowanie algorytmu ewolucyjnego

2.1 Zasada działania algorytmów ewolucyjnych

Dlaczego działają, kiedy nie

2.2 Schemat algorytmu

Programy ewolucyjne są zbudowane według poniższego ogólnego schematu [Michalewicz Zbigniew, Algorytmy genetyczne …] gdzie P(t) oznacza populację w iteracji t.

```
procedure program ewolucyjny
begin
  t←0
  ustal początkowe P(t)
  oceń P(t)
  while (not warunek zakończenia) do
  begin
    t←t+1
    wybierz P(t) z P(t-1)  //Selekcja
    zmień P(t)             //Mutacja oraz krzyżowanie
    oceń P(t)
  end
end
```

W tej pracy przedstawię jak przystosowałem powyższy schemat do przedstawionego problemu.

2.3 Populacja początkowa

Populację początkową można wygenerować jako populację rozwiązań zerowych czyli takich bez jakiejkolwiek rozpoczętej uprawy w rozwiązaniu. Jest to jednak nie najlepszy sposób z powodu tego że część iteracji algorytmu na początku będzie musiała być poświęcona na wstępne zapełnienie rozwiązań. Można ten czas oszczędzić generując od razu populację w której każde rozwiązanie ma już kilka zaczętych upraw. Przykład takiej metody jest przedstawiony poniżej

Dla każdego rozwiązania w populacji losujemy kolejność pól zgodnie z którą będzie uzupełniane rozwiązanie. Następnie dla każdego pola losujemy uprawę oraz dzień rozpoczęcia z przedziału dopuszczalnego dla danej uprawy. Jeśli po sprawdzeniu ograniczeń zasobowych rozwiązanie z nową uprawą jest dopuszczalne to dodajemy tą uprawę, jeśli nie to przechodzimy do następnego pola w kolejności.

Plusy tej metody są następujące żadne pole ani rodzaj uprawy nie będzie faworyzowany oraz otrzymamy zawsze rozwiązanie dopuszczalne. Minusem jest to że metoda jest bardziej

skomplikowana od generacji rozwiązań zerowych oraz to że metoda stworzy co najwyżej jedną uprawę na polu.

2.4 Selekcja

Selekcja polega na wyborze rozwiązań które przechodzą do następnej iteracji algorytmu. Celem tego etapu jest wprowadzenie presji ewolucyjnej. Sprawia ona że rozwiązania dążą do poprawy funkcji celu. Bez selekcji populacja rozwiązań losowo błądziłaby po przestrzeni rozwiązań. W metodach selekcji opisanych poniżej lepsze rozwiązania mają większe prawdopodobieństwo przejścia do następnej iteracji.

Metoda koła ruletki <todo>

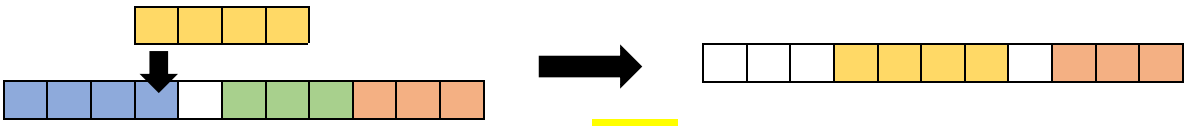
Selekcja rankingowa <todo>

Selekcja turniejowa <todo>

2.5 Mutacja

Mutacja w algorytmach ewolucyjnych ma na celu zróżnicowanie rozwiązań i przeciwdziała utknięciu populacji w lokalnym ekstremum. Polega ona na małej zmianie losowej części rozwiązania.

Proponowana metoda mutacji polega na losowym wyborze typu uprawy, następnie dnia rozpoczęcia z możliwych dla danego typu. Należy też wybrać pole na którym zostanie rozpoczęta ta uprawa. Można wybrać pole losowo gdzie każde pola ma równe prawdopodobieństwo, lub uwzględnić współczynnik dopasowania pola do typu produktu. Dobrze nadaje się do tego do tego metoda ruletki w której prawdopodobieństwo wyboru jest wprost proporcjonalne do współczynnika dopasowania. Jeśli na danym polu i terminie są już uprawy to należy je usunąć z rozwiązania.



Rys. 1 <todo>

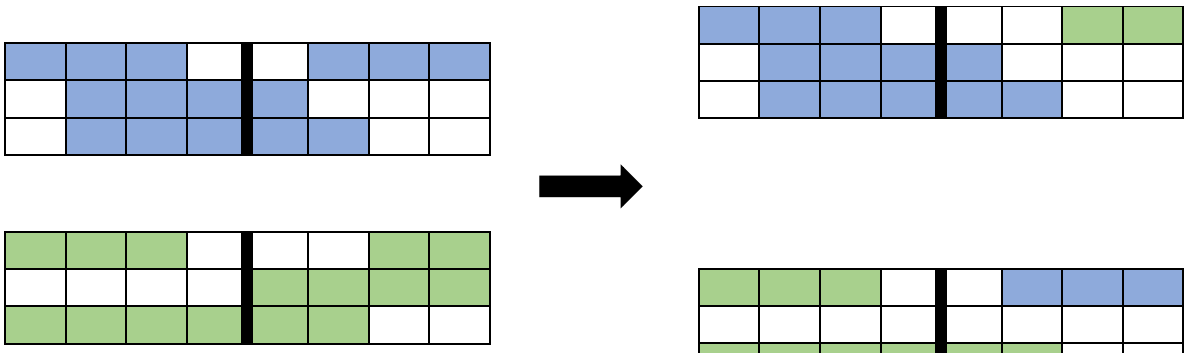
2.6 Krzyżowanie

Krzyżowanie ma za zadanie <todo>

W przedstawionym problemie nasuwają się dwa główne sposoby na przeprowadzenie krzyżowania. Krzyżowanie ze względu na:

- Dni
- Pola

Krzyżowanie ze względu na dni polega na wyborze części dni z jednego rozwiązania i reszty z drugiego. Problemem który pojawi się podczas przeprowadzania tego typu krzyżowania są uprawy będące w trakcie na granicy łączenia. Aby zminimalizować ten problem zastosowano krzyżowanie jednopunktowe z punktem podziału w połowie rozwiązania.



Rys. 2 <todo>

Krzyżowanie ze względu na pola jest prostsze ponieważ kolejność pól nie ma znaczenia. Wybrano metodę krzyżowania biorącą parzyste pola z pierwszego rozwiązania oraz nieparzyste z drugiego i tworzące w ten sposób nowe rozwiązanie. Aby utrzymać symetrię należy powtórzyć proces zamieniając najpierw kolejnością rozwiązania, wtedy uzyskamy drugie nowe rozwiązanie.

2.7 Ocena rozwiązania i funkcja kary

Na skutek krzyżowania lub mutacji mogą się pojawić rozwiązanie nie spełniające ograniczeń zasobowych. Inne ograniczenia będą zawsze spełnione. Można ten problem rozwiązać stosując funkcję kary lub funkcję poprawy.

Funkcja kary polega na

Jest ona stosowana nie tylko w algorytmach ewolucyjnych, ale też w innych metodach optymalizacji.

Wartoś funkcji kary zależy od ilości przekroczonych ograniczeń oraz stopnia ich przekroczenia. W przedstawionym problemie zastosowano

Funkcja kary może być zależna od numeru iteracji. Z reguły wartość funkcji kary rośnie wraz z kolejnymi iteracjami. Ma to na celu sprawienie aby na początku działania algorytmu funkcja kary nie wpływała znacząco na wartość funkcji celu, ale wraz z czasem coraz bardziej. Pozwala to na przeszukanie zbioru rozwiązań niedopuszczalnych a jednocześnie

2.8 Parametry algorytmu oraz warunek zakończenia

W opracowanym algorytmie jest możliwość wyboru rodzaju populacji początkowej, krzyżowania, mutacji, funkcji kary oraz selekcji. Dodatkowo można określić prawdopodobieństwo mutacji.

Algorytm zakończy pracę kiedy wystąpi jeden z poniższych warunków

- Maksymalna liczba iteracji zostanie przekroczona.
- Maksymalna liczba iteracji bez poprawy wartości funkcji celu zostanie przekroczona.
- Nastąpi manualne zatrzymanie algorytmu.

3 Implementacja programu

3.1 Struktury danych

Opisy najważniejszych klas, opis implementacji postaci rozwiązania

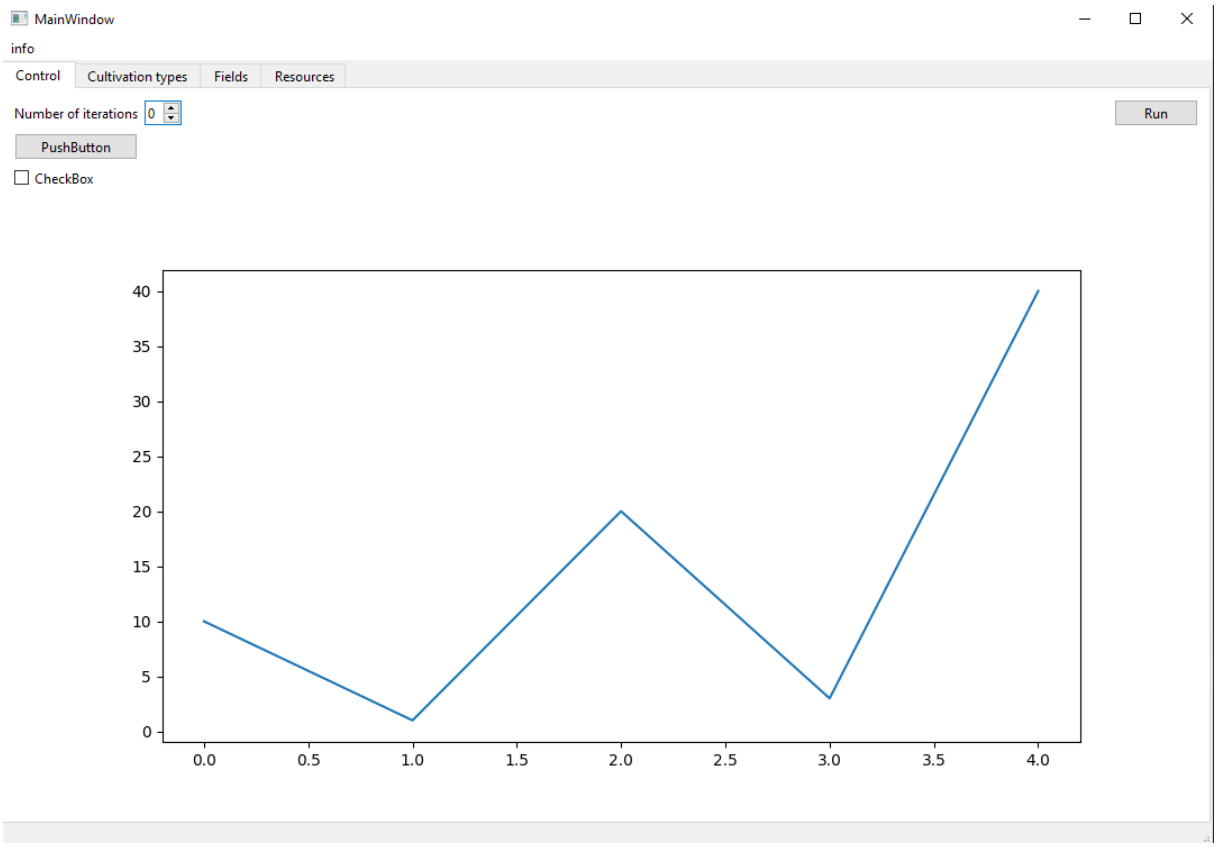
3.2 Implementacja mutacji, selekcji i krzyżowania

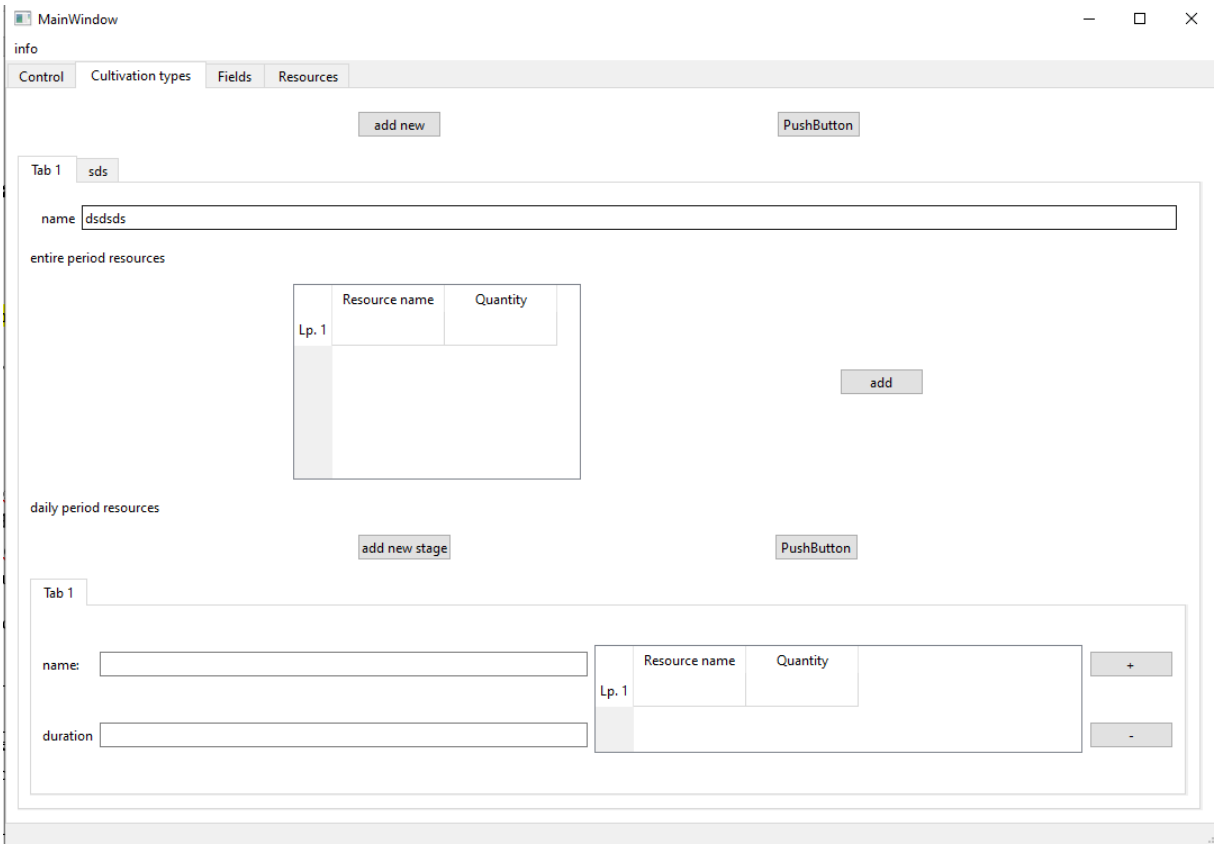
3.3 Implementacja interfejsu graficznego

Zdecydowałem się na użycie Pythona również do utworzenia interfejsu graficznego. Nie jest to typowo język wykorzystywany do tego celu, ale taki wybór pozwala wykorzystać jego biblioteki jak między innymi matplotlib i pandas do tworzenia wykresów. Dodatkową zaletą wykorzystania jednego języka jest uproszczenie procesu budowania i instalacji programu.

Dwa najpopularniejsze narzędzia do tworzenia interfejsu graficznego w pythonie to Tkinter oraz Qt.

	Tkinter	Qt
Instalacja	Nie konieczna, jest to standardowa biblioteka Pythona	Konieczna
Prosty do zrozumienia	Tak, mała ilość funkcjonalności	Nie, zrozumienie całej biblioteki zajmuje dużo czasu
Dostępne widżety	Tylko podstawowe widżety	Szeroki wybór
Graficzny konstruktor interfejsu	Brak	Dostępne narzędzie Qt Designer pozwalające dodawać widgety metodą przeciągnij i upuść (ang. drag and drop) oraz wygenerować z tego kod Pythona.
Licencja		


MainWindow
info
Control
Cultivation types
Fields
Resources
Number of iterations
0
Run
PushButton
CheckBox
40
35
30
25
20
15
10
5
0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0


MainWindow
info
Control
Cultivation types
Fields
Resources
add new
PushButton
Tab 1
sds
name
dsdsds
entire period resources
Resource name
Quantity
Lp. 1
add
daily period resources
add new stage
PushButton
Tab 1
name:
duration
Resource name
Quantity
Lp. 1
+
-

4 Testy programu

4.1 <TODO>

5 Podsumowanie

6 Literatura

[] Michalewicz Zbigniew, Algorytmy genetyczne + struktury danych = programy ewolucyjne z angielskiego przełożył Zbigniew Nahorski.Wydanie trzecie. Warszawa : Wydawnictwa Naukowo-Techniczne, https://katalogagh.cyfronet.pl/lib/item?id=chamo:53931&fromLocationLink=false&theme=bg agh

[] Marcin Klimek, Predyktywno-reaktywne harmonogramowanie produkcji z ograniczoną dostępnością zasobów rozprawa doktorska mgr inż.Wydawnictwo AGH http://winntbg.bg.agh.edu.pl/rozprawy2/10260/full10260.pdf

[] Ibrahim M. Al-Harkan, et al, A Decision Support System for Optimal Use of Irrigation Water and Crop Selection, Journal of King Saud University - Engineering Sciences, Volume 21, Issue 2, 2009, Pages 77-84, ISSN 1018-3639, https://doi.org/10.1016/S1018-3639(18)30511-7

[]Ya Ivanyo et al 2021 J. Phys.: Conf. Ser. 1989 012041 Optimization models of agricultural production with heterogeneous land resources https://doi.org/10.1088/1742-6596/1989/1/012041

[]Ya M Ivanyo et al 2022 IOP Conf. Ser.: Earth Environ. Sci. 988 022083 Ecological-Mathematical Modeling in Planning Production of Agricultural Products in Conditions of Risks http://dx.doi.org/10.1088/1755-1315/988/2/022083

[] Urja Thakkar, et al, Application of Operations Research in Agriculture https://www.ijser.org/researchpaper/Application-of-Operations-Research-in-Agriculture.pdf

[]Bautista-Valhondo, Joaquín & García, Amaia & Suarez, Raul & Mateo, Manuel & Pastor, Rafael & Corominas, Albert. (1999). Application of genetic algorithms to assembly sequence planning with limited resources. Proceedings of the IEEE International Symposium on

Assembly and Task Planning. 411 - 416.
https://www.researchgate.net/publication/3811577_Application_of_genetic_algorithms_to_assembly_sequence_planning_with_limited_resources

[]PYPL PopularitY of Programming Language Index
https://pypl.github.io/PYPL.html

[]Tkinker - narzędzie do tworzenia GUI
https://docs.python.org/3/library/tkinter.html

[]Maffezzoli, et al, (2022). Agriculture 4.0: A systematic literature review on the paradigm, technologies and benefits. Futures. 142. 102998. 10.1016/j.futures.2022.102998.
https://www.researchgate.net/publication/362191241_Agriculture_40_a_systematic_literature_review_on_the_paradigm_technologies_and_benefits