چکیده

مسـئله‌ی N-وزیر یکی از مسـائل کلاسـیک در ترکیبیات و علوم کامپیوتر اسـت. روش‌ها و الگوریتم‌های متعددی برای حل این مسئله پیشنهاد شده است که شاید مشهورترین آن الگوریتم عقبگرد باشد. یکی دیگر از روش‌هایی که در منابع معتبر برای این مسئله پیشنهاد شده است استفاده از الگوریتم تکاملی است. در این گزارش با استفاده از همین منابع معتبر و اندکی تغییر، با استفاده از الگوریتم ژنتیک پاسخ مسئله به دست آورده می‌شود و نتایج آن بررسی می‌شود.

واژه‌های کلیدی:

N-وزیر، الگوریتم تکاملی، الگوریتم ژنتیک

فهرست مطالب

صفحه

۱

فصل اول: مقدمه

مسـئله‌ی N-وزیر[1] یکی از مسـائل کلاسـیک در ترکیبیات و علوم کامپیوتر اسـت. در این مسـئله یک صفحه‌ی شطرنج N×N در نظر گرفته شده و N وزیر را باید طوری در این صفحه قرار داد که هیچ کدام دیگری را تهدید نکند.

یکی از ساده‌ترین روش‌ها برای حل این مسئله الگوریتم عقبگرد[2] است. در این روش در واقع با شروع از خانه‌ی اول و امتحان کردن تمام حالت‌ها به چینش نهایی وزیرها می‌رسیم. به وضوح پیچیدگی زمانی این روش به شدت بالاست، چون در هر چینش باید وضعیت تهدید کردن چند جفت مهره بررسی شود و در هر بار مواجهه با چینش نادرست، دوباره به خانه‌ی قبلی برگشته و به صورت بازگشتی این کار ادامه پیدا کند.

شاید چنین به نظر برسد که با روش‌های فراابتکاری مثل فیلترینگ در CSP بتوان زمان محاسبات را بهبود داد، ولی همچنان باید حالت‌های بسـیار زیادی بررسـی شـوند و حتی با وجود فیلترینگ هم، اگر الگوریتم بخواهد به بررسـی تهدید جفت‌های وزیرها بپردازد به زمان زیادی نیاز خواهد داشـت و تفاوتی میان روش عقبگرد و این روش دیده نخواهد شد.

یکی دیگر از روش‌ها که در (Evolutionary algorithms, 2020) آمده روش الگوریتم تکاملی[3] است که از جمله الگوریتم‌هایی هستند که از طبیعت الهام گرفته شده‌اند. در چنین روشی هر چینش از صفحه‌ی شطرنج را به صورت یک رشته نمایش می‌دهیم که در واقع نمایانگر یک وضعیت است. جمعیت[4] را مجموعه‌ای از این رشـته‌ها در نظر گرفته که هر جمعیت نماینده‌ی یک نسـل اسـت. با بهره گرفتن از مبانی ژنتیک، در اینجا آن دسته از اعضای جمعیت که نسبت به بقیه ارزش بیشتری دارند با هم طبق قاعده‌ی مشخصی ترکیب شده که این مشابه تولید مثل کردن در طبیعت است. باز هم مشابه مبانی ژنتیک، گاهی اوقات جهش‌هایی در رشته‌ها دیده می‌شود. این تولید نسل‌ها تا حد معینی ادامه پیدا کرده و یا تا جایی پیش می‌رود که الگوریتم به جواب مطلوب با امتیاز بیشینه که همان جواب است برسد.

---

[1] N-queen

[2] backtracking

[3] evolutionary algorithm

[4] population

۲ فصل دوم: شرح الگوریتم ژنتیک در مسئله‌ی N-وزیر و پیاده‌سازی آن

در این فصل به شرح الگوریتم ژنتیک و پیاده سازی آن در مسئله‌ی N-وزیر پرداخته می شود. پیاده سازی این الگوریتم به زبان پایتون و در یک مخزن خصوصی گیت‌هاب صورت گرفته است.

۱-۲-کلیت الگوریتم و تولید نسل‌ها

قبل از هر چیز به یک جمعیت اولیه نیاز است که در واقع همان نسل اول را تشکیل می‌دهد. این جمعیت با بهره گرفتن از کلاس `Population` ساخته می‌شود. اعضای یک جمعیت کروموزوم‌ها هستند که از کلاس `Chromosome` ایجاد می‌شوند. کلاس `Chromosome` اطلاعات مربوط به هر کروموزوم از جمله رشته‌ی مربوط به آن کروموزوم را ذخیره می‌کند. باید توجه داشت که در بسیاری از منابع رشته‌ها به صورت ستونی پیاده شده‌اند، اما در اینجا نمایش رشته‌ها بیانگیر جایگاه سطری هر وزیر است؛ به عنوان مثال رشته‌ی `[2, 4, 3, 1, 0]` نماینده‌ی چینشی است که وزیرها در ستون صفرم و سطر دوم، ستون اول و سطر چهارم، ستون دوم و سطر سوم، ستون سوم و سطر اول و ستون چهارم و سطر صفرم قرار گرفته‌اند. همچنین از کتابخانه‌ی numpy برای ذخیره سازی رشته‌ها استفاده شده که سریع‌تر و بهینه‌تر است.

قبل از هر چیز حدی برای تولید نسل‌ها مشخص می‌کنیم، که این حد در پیاده سازی به صورت پیش فرض ۳۰۰ گرفته شده است؛ یعنی الگوریتم حداکثر تا ۳۰۰ نسل جلو رفته و اگر یکی از اعضای هر کدام از نسل‌ها بیشینه‌ی امتیاز را داشت، آن عضو را به عنوان پاسخ انتخاب می‌کند. این همان منطقی است که در تابع `main()` پیاده‌سازی شده است. معیار امتیازدهی در ادامه مشخص خواهد شد.

هر نسلی برای آنکه به نسل بعدی تبدیل شود از سه مرحله‌ی اصلی عبور می‌کند؛ مرحله‌ی اول بازترکیبی[۱]، مرحله‌ی دوم انتخاب[۲] و مرحله‌ی سوم جهش[۳] است که در ادامه هر کدام از این مراحل شرح داده می‌شوند.

---

۱ recombination

۲ selection

۳ mutation

۲-۲-بازترکیبی

این قسمت همان قسمت تولید مثل است که در واقع جمعیت یک نسل با هم مطابق با سیاست خاصی ترکیب شده و فرزندانی را به وجود می‌آورند. مرسوم‌ترین روش به این صورت است که اعضای جمعیت به صورت جفت جفت با هم ترکیب شده و فرزندانی را به وجود آورند، یعنی عدد ترکیبی[۱] مقدار $\rho = 2$ داشته باشد. باید توجه داشت که ρ مقادیر دیگری هم می‌تواند بگیرد، که به عنوان مثال مقدار ۱ به معنای مشارکت تکی یک عضو در تولید مثل است که در طبیعت به صورت تولید مثل غیرجنسی[۲] دیده می‌شود. مقادیر بیشتر از ۲ هم می‌توان استفاده کرد که چندان مرسوم نیستند.

تولید مثل یا همان ترکیب شدن رشته‌ها (کروموزوم‌ها) با هم به این صورت است که به صورت تصادفی یک حد جداکننده انتخاب شده و دو رشته از همان حد، هر کدام به دو زیر رشته تقسیم می شوند. آنگاه زیررشته‌ی اول حاصل از رشته‌ی اول با زیررشته‌ی دوم حاصل از رشته‌ی دوم ترکیب شده و فرزند اول را شکل می‌دهد، و همچنین زیررشته‌ی اول حاصل از رشته‌ی دوم با زیررشته‌ی دوم حاصل از رشته‌ی اول ترکیب شده و فرزند دوم را تشکیل می‌دهد. پس نتیجه‌ی ترکیب دو عضو از یک نسل، دو فرزند جدید است. این ترکیب شدن‌ها توسط تابع `reproduce(other)` صورت گرفته که در واقع `self` با `other` که هر دو از کلاس `Chromosome` هستند ترکیب می‌شوند.

۳-۲-انتخاب

پس از مرحله‌ی بازترکیبی، الگوریتم وارد مرحله‌ی انتخاب می‌شود که در اینجا به تعداد اندازه‌ی جمعیت که در کلاس `Population` با متغیر `population_size` مشخص می شود، از جمعیت کنونی که شامل فرزندان متولدشده در مرحله‌ی قبل هم هستند، رشته‌های برتر، یعنی با امتیاز بیشتر برگزیده می‌شوند.

---

[۱] mixing number

[۲] asexual reproduction

تابع `()fitness` در کلاس `Chromosome` تعریف شـــده که برای هر موجودی از این کلاس یک امتیاز را محاســـبه می‌کند. این امتیاز طبق (`Evolutionary algorithms, 2020`) برای هر کروموزوم به صـــورت تعداد جفت وزیرهایی که همدیگر را تهدید نمی‌کنند تقســـیم بر تعداد کل جفت وزیرهای ممکن، یعنی $\binom{N}{2}$ محاسبه می‌شود. چون نمایش رشته‌ها به صورت سطری است، هیچ دو وزیری نمی‌توانند به صورت ستونی همدیگر را تهدید کنند. بنابراین دو حالت دیگر، یعنی تهدید ستونی و مورب آزموده می شود که در نو شتن آن از (`Ramin Saljoughinejad`, بدون تاریخ) ا ستفاده شده ا ست. تابع `()solution_found` هم که در کلاس `Population` تعریف شده کروموزومی را پیدا می‌کند که بیشترین امتیاز، یعنی امتیاز ۱ را داشته باشد و در واقع همان کروموزوم به عنوان جواب انتخاب می‌شود.

۴-۲-جهش

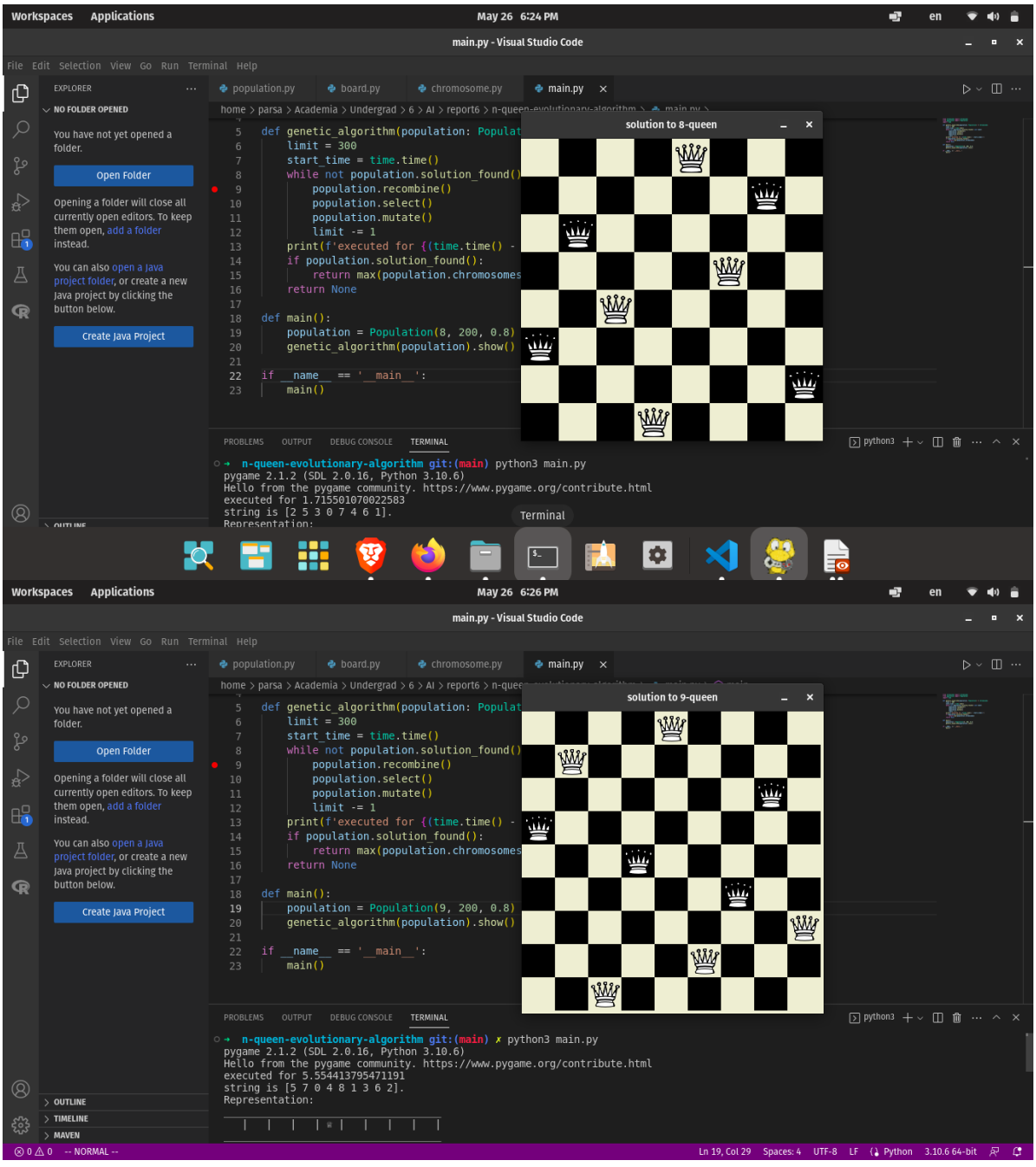
پس از مرحله‌ی انتخاب، یک نمونه از جمعیت انتخاب شـــده که یکی از ژن‌های هر کدام از کروموزوم‌های نمونه‌ی انتخاب شده دچار جهش می‌شود؛ به این معنی که آن ژن به صورت تصادفی با یکی از اعداد بین ۰ تا $N - 1$ جایگزین می‌شود. اندازه‌ی نمونه هم توسط متغیر `mutation_rate` مشخص می‌شود.

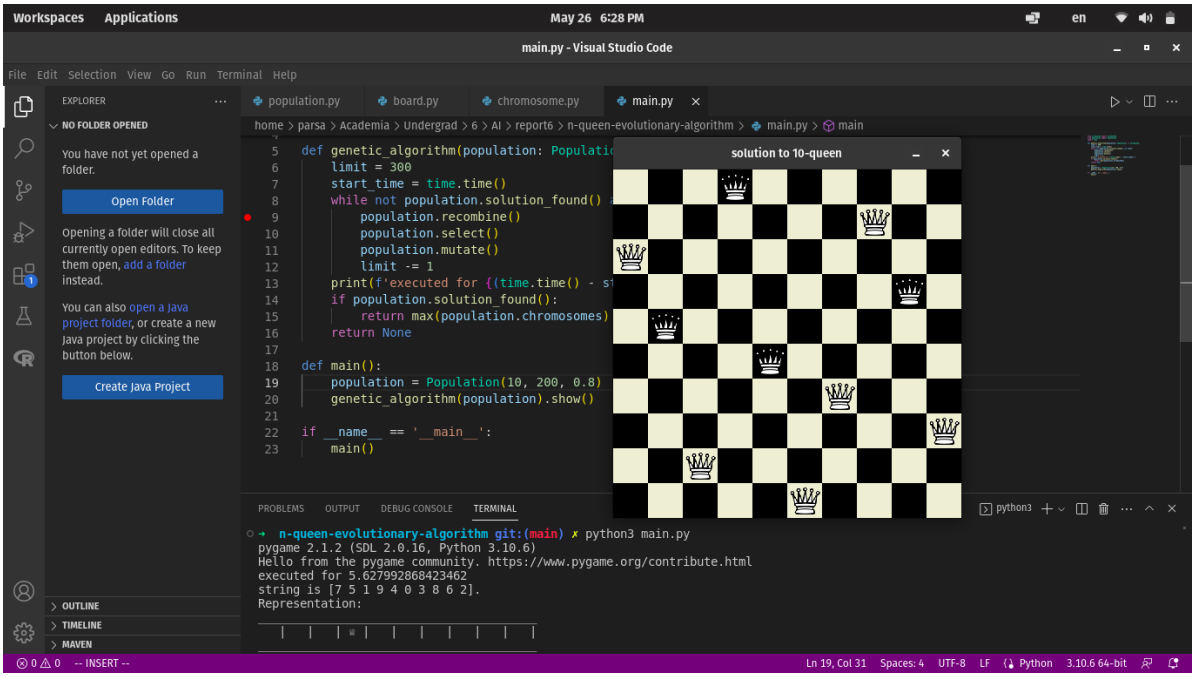
۳

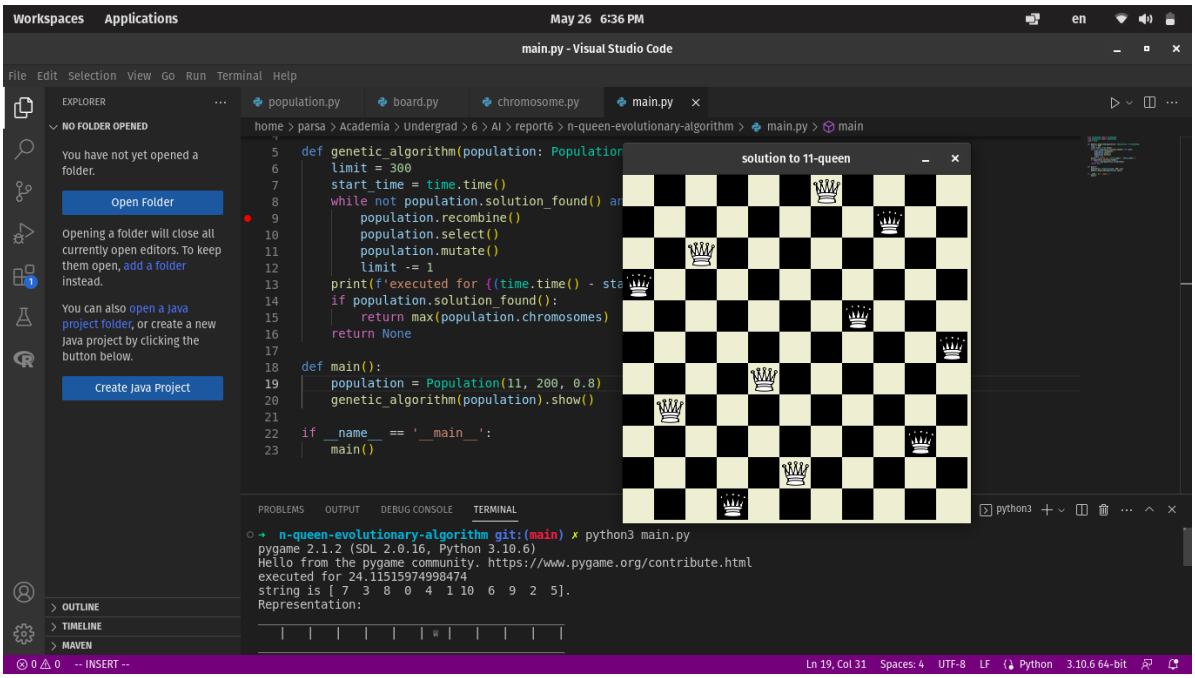
فصل سوم: جمع‌بندی و نتیجه‌گیری

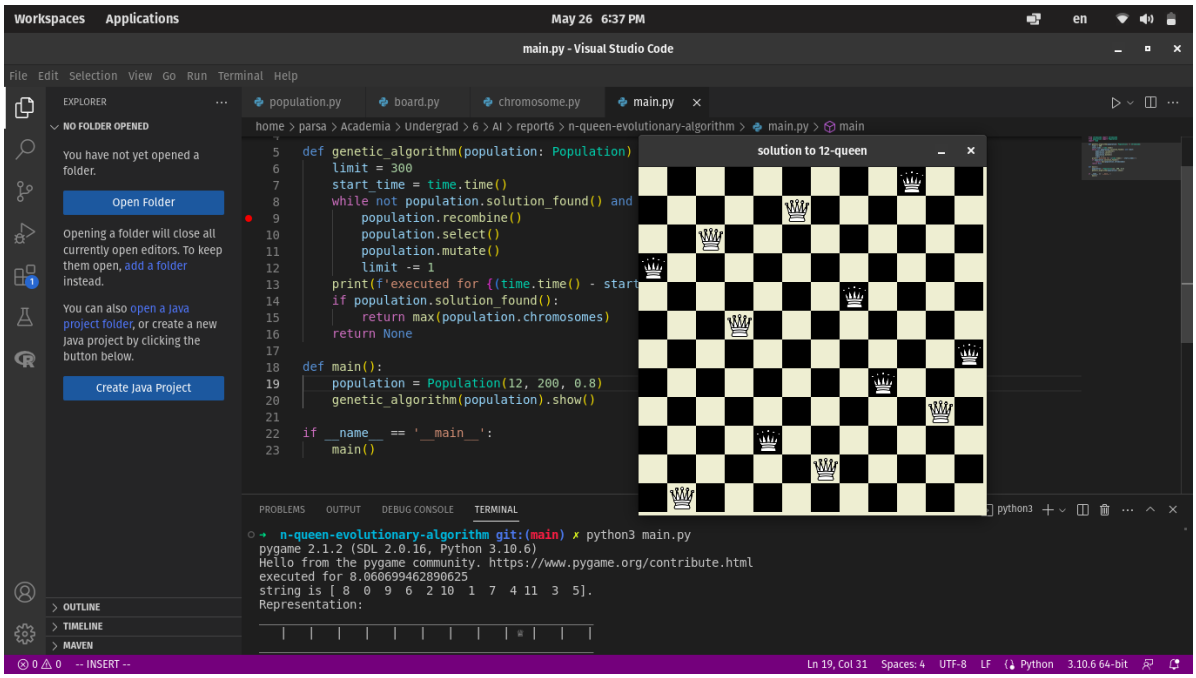
در این قسمت الگوریتم و زمان اجرای آن را برای مقادیر N = 8 تا N = 16 بررسی می‌کنیم. باید توجه کرد که Nهای بزرگتر لزوماً به معنای زمان اجرای بیشتر نیست. می‌توان گفت با اینکه با افزایش مقدار N به طور میانگین زمان اجرا بیشتر می‌شود، اما ممکن است به دلیل طبیعت تصادفی الگوریتم ژنتیک، سریع‌تر پاسخ دریافت شود. همچنین ممکن است یک بار اجرا به پاسخ منتهی نشود که باید دوباره امتحان کرد. اما باید توجه داشت که به ازای N = 1 جواب مسئله بدیهی و به ازای N = 2 و N = 3 مسئله فاقد جواب است.

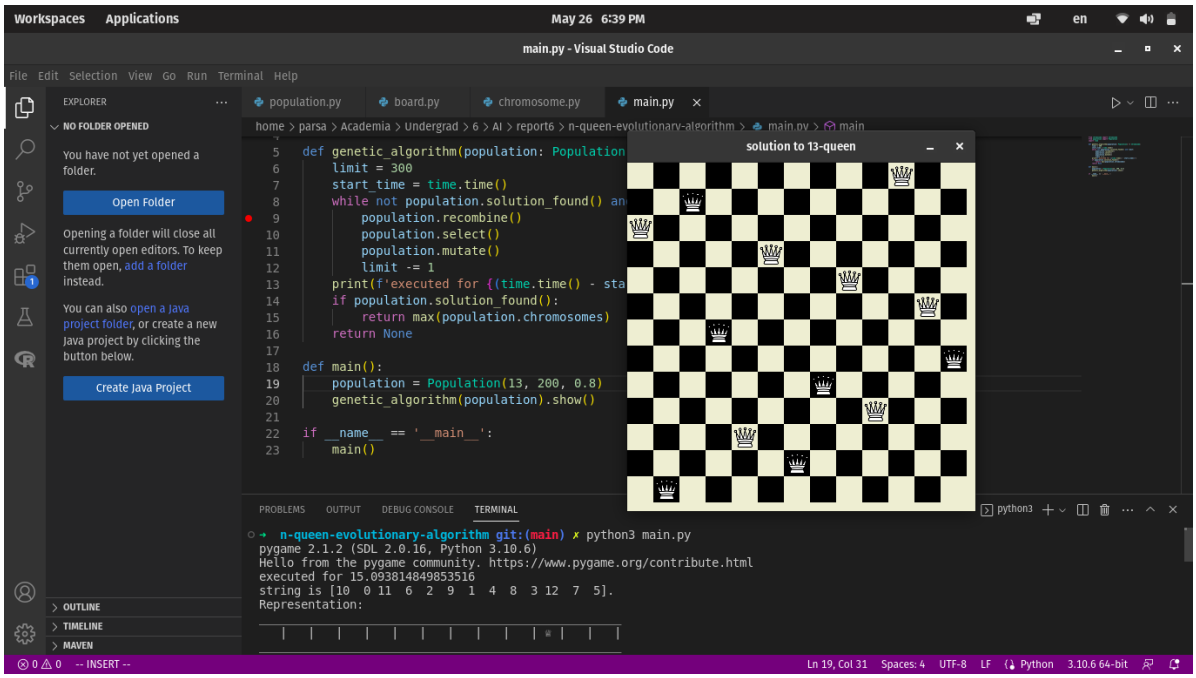
شکل‌های زیر حل مسائل ۸-وزیر تا ۱۶-وزیر را همراه با زمان حل نشان می‌دهند. در کنسول هم یک ماتریس به همراه زمان اجرا چاپ می‌شود، اما صفحه‌ی گرافیکی با تغییر کد (Rajesh-Reddy1/Sample-CHESS-board, n.d.) و با استفاده از ماژول PyGame ایجاد شده است.

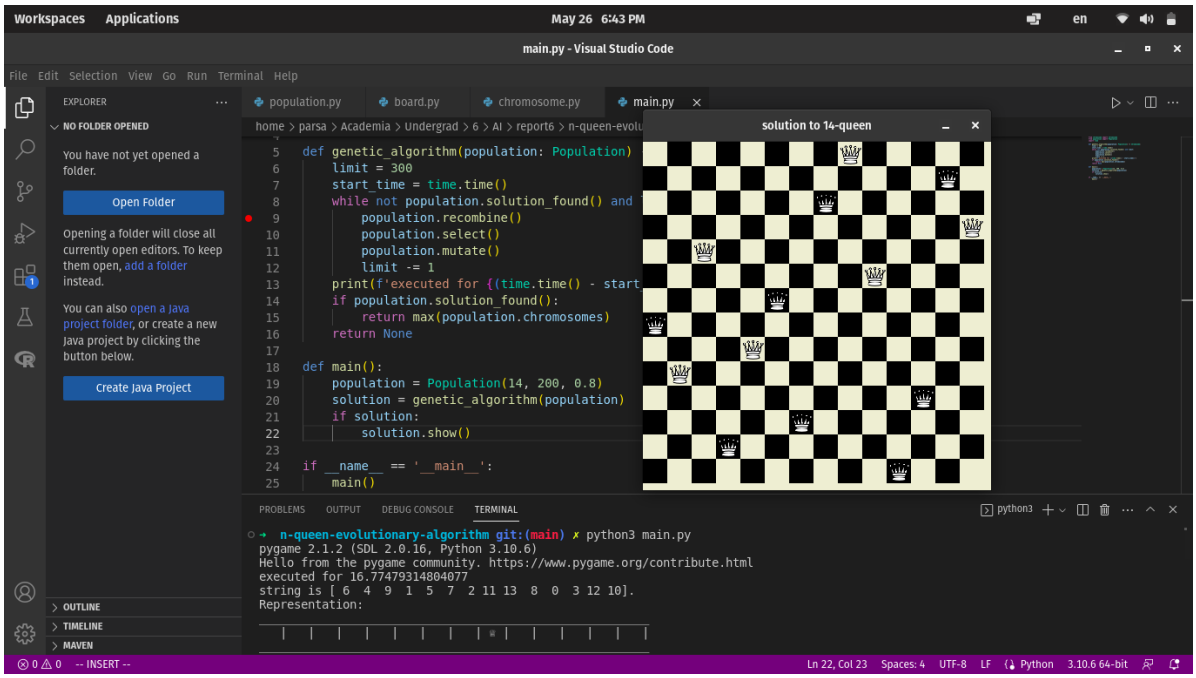

Workspaces Applications
May 26 6:24 PM
en
main.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
You can also open a Java project folder, or create a new Java project by clicking the button below.
Create Java Project
population.py
board.py
chromosome.py
main.py
solution to 8-queen
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
n-queen-evolutionary-algorithm git:(main) python3 main.py
pygame 2.1.2 (SDL 2.0.16, Python 3.10.6)
Hello from the pygame community. https://www.pygame.org/contribute.html
executed for 1.715501070022583
string is [2 5 3 0 7 4 6 1].
Representation:
Terminal
Workspaces Applications
May 26 6:26 PM
en
main.py - Visual Studio Code
solution to 9-queen
n-queen-evolutionary-algorithm git:(main) ✗ python3 main.py
pygame 2.1.2 (SDL 2.0.16, Python 3.10.6)
Hello from the pygame community. https://www.pygame.org/contribute.html
executed for 5.554413795471191
string is [5 7 0 4 8 1 3 6 2].
Representation:
OUTLINE
TIMELINE
MAVEN
Ln 19, Col 29 Spaces: 4 UTF-8 LF Python 3.10.6 64-bit

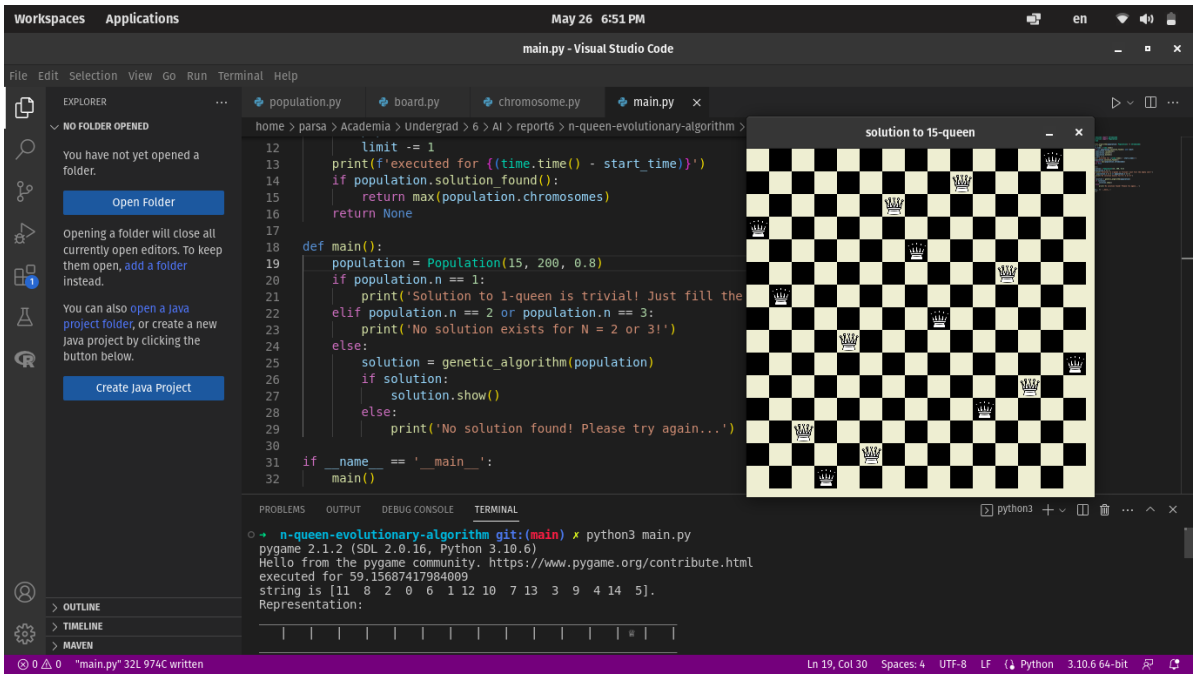

Workspaces Applications
May 26 6:28 PM
en
main.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
You can also open a Java project folder, or create a new Java project by clicking the button below.
Create Java Project
population.py
board.py
chromosome.py
main.py
home > parsa > Academia > Undergrad > 6 > AI > report6 > n-queen-evolutionary-algorithm > main.py > main
def genetic_algorithm(population: Populati
limit = 300
start_time = time.time()
while not population.solution_found()
population.recombine()
population.select()
population.mutate()
limit -= 1
print(f'executed for {(time.time() - st
if population.solution_found():
return max(population.chromosomes)
return None
def main():
population = Population(10, 200, 0.8)
genetic_algorithm(population).show()
if __name__ == '__main__':
main()
solution to 10-queen
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
python3
n-queen-evolutionary-algorithm git:(main) ✗ python3 main.py
pygame 2.1.2 (SDL 2.0.16, Python 3.10.6)
Hello from the pygame community. https://www.pygame.org/contribute.html
executed for 5.627992868423462
string is [7 5 1 9 4 0 3 8 6 2].
Representation:
OUTLINE
TIMELINE
MAVEN
Ln 19, Col 31 Spaces: 4 UTF-8 LF Python 3.10.6 64-bit
-- INSERT --

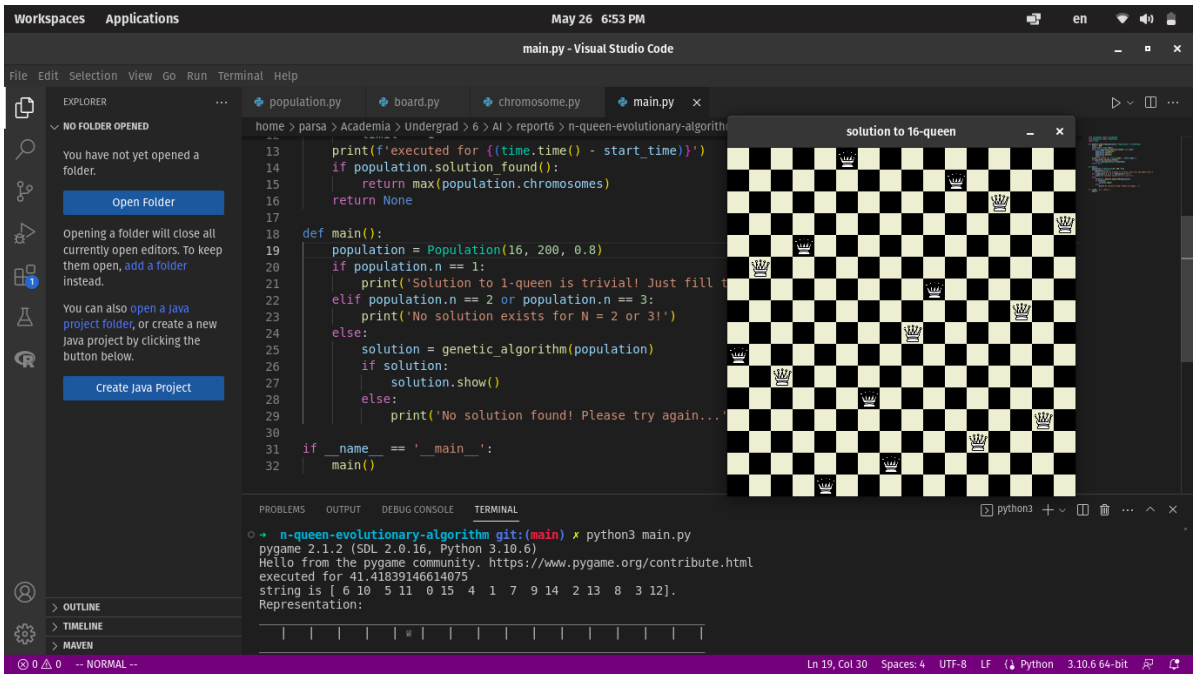

Workspaces Applications
May 26 6:36 PM
en
main.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
You can also open a Java project folder, or create a new Java project by clicking the button below.
Create Java Project
population.py
board.py
chromosome.py
main.py
home > parsa > Academia > Undergrad > 6 > AI > report6 > n-queen-evolutionary-algorithm > main.py > main
def genetic_algorithm(population: Populatior
limit = 300
start_time = time.time()
while not population.solution_found() a
population.recombine()
population.select()
population.mutate()
limit -= 1
print(f'executed for {(time.time() - sta
if population.solution_found():
return max(population.chromosomes)
return None
def main():
population = Population(11, 200, 0.8)
genetic_algorithm(population).show()
if __name__ == '__main__':
main()
solution to 11-queen
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
python3
n-queen-evolutionary-algorithm git:(main) ✗ python3 main.py
pygame 2.1.2 (SDL 2.0.16, Python 3.10.6)
Hello from the pygame community. https://www.pygame.org/contribute.html
executed for 24.115159749984 74
string is [7 3 8 0 4 1 10 6 9 2 5].
Representation:
OUTLINE
TIMELINE
MAVEN
Ln 19, Col 31 Spaces: 4 UTF-8 LF Python 3.10.6 64-bit
-- INSERT --


Workspaces Applications
May 26 6:37 PM
en
main.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
You can also open a Java project folder, or create a new Java project by clicking the button below.
Create Java Project
population.py
board.py
chromosome.py
main.py
home > parsa > Academia > Undergrad > 6 > AI > report6 > n-queen-evolutionary-algorithm > main.py > main
def genetic_algorithm(population: Population)
limit = 300
start_time = time.time()
while not population.solution_found() and
population.recombine()
population.select()
population.mutate()
limit -= 1
print(f'executed for {(time.time() - start
if population.solution_found():
return max(population.chromosomes)
return None
def main():
population = Population(12, 200, 0.8)
genetic_algorithm(population).show()
if __name__ == '__main__':
main()
solution to 12-queen
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
n-queen-evolutionary-algorithm git:(main) ✗ python3 main.py
pygame 2.1.2 (SDL 2.0.16, Python 3.10.6)
Hello from the pygame community. https://www.pygame.org/contribute.html
executed for 8.060699462890625
string is [8 0 9 6 2 10 1 7 4 11 3 5].
Representation:
OUTLINE
TIMELINE
MAVEN
Ln 19, Col 31 Spaces: 4 UTF-8 LF Python 3.10.6 64-bit


Workspaces Applications
May 26 6:39 PM
en
main.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
You can also open a Java project folder, or create a new Java project by clicking the button below.
Create Java Project
population.py
board.py
chromosome.py
main.py
home > parsa > Academia > Undergrad > 6 > AI > report6 > n-queen-evolutionary-algorithm > main.py > main
def genetic_algorithm(population: Population
limit = 300
start_time = time.time()
while not population.solution_found() an
population.recombine()
population.select()
population.mutate()
limit -= 1
print(f'executed for {(time.time() - sta
if population.solution_found():
return max(population.chromosomes)
return None
def main():
population = Population(13, 200, 0.8)
genetic_algorithm(population).show()
if __name__ == '__main__':
main()
solution to 13-queen
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
n-queen-evolutionary-algorithm git:(main) ✗ python3 main.py
pygame 2.1.2 (SDL 2.0.16, Python 3.10.6)
Hello from the pygame community. https://www.pygame.org/contribute.html
executed for 15.093814849853516
string is [10 0 11 6 2 9 1 4 8 3 12 7 5].
Representation:
OUTLINE
TIMELINE
MAVEN
Ln 19, Col 31 Spaces: 4 UTF-8 LF Python 3.10.6 64-bit


Workspaces Applications
May 26 6:43 PM
en
main.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
You can also open a Java project folder, or create a new Java project by clicking the button below.
Create Java Project
OUTLINE
TIMELINE
MAVEN
population.py
board.py
chromosome.py
main.py
solution to 14-queen
def genetic_algorithm(population: Population)
limit = 300
start_time = time.time()
while not population.solution_found() and
population.recombine()
population.select()
population.mutate()
limit -= 1
print(f'executed for {(time.time() - start
if population.solution_found():
return max(population.chromosomes)
return None
def main():
population = Population(14, 200, 0.8)
solution = genetic_algorithm(population)
if solution:
solution.show()
if __name__ == '__main__':
main()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
n-queen-evolutionary-algorithm git:(main) ✗ python3 main.py
pygame 2.1.2 (SDL 2.0.16, Python 3.10.6)
Hello from the pygame community. https://www.pygame.org/contribute.html
executed for 16.77479314804077
string is [6 4 9 1 5 7 2 11 13 8 0 3 12 10].
Representation:
Ln 22, Col 23 Spaces: 4 UTF-8 LF Python 3.10.6 64-bit
-- INSERT --


Workspaces Applications
May 26 6:51 PM
en
main.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
EXPLORER
NO FOLDER OPENED
You have not yet opened a folder.
Open Folder
Opening a folder will close all currently open editors. To keep them open, add a folder instead.
You can also open a Java project folder, or create a new Java project by clicking the button below.
Create Java Project
OUTLINE
TIMELINE
MAVEN
population.py
board.py
chromosome.py
main.py
solution to 15-queen
limit -= 1
print(f'executed for {(time.time() - start_time)}')
if population.solution_found():
return max(population.chromosomes)
return None
def main():
population = Population(15, 200, 0.8)
if population.n == 1:
print('Solution to 1-queen is trivial! Just fill the
elif population.n == 2 or population.n == 3:
print('No solution exists for N = 2 or 3!')
else:
solution = genetic_algorithm(population)
if solution:
solution.show()
else:
print('No solution found! Please try again...')
if __name__ == '__main__':
main()
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
n-queen-evolutionary-algorithm git:(main) ✗ python3 main.py
pygame 2.1.2 (SDL 2.0.16, Python 3.10.6)
Hello from the pygame community. https://www.pygame.org/contribute.html
executed for 59.15687417984009
string is [11 8 2 0 6 1 12 10 7 13 3 9 4 14 5].
Representation:
Ln 19, Col 30 Spaces: 4 UTF-8 LF Python 3.10.6 64-bit
"main.py" 32L 974C written



همان طور که مشاهده می‌شود کمینه زمان مربوط به ۸-وزیر و حدود دو ثانیه و بیشینه زمان مربوط به ۱۵-وزیر و حدود یک دقیقه است.

پس راهکاری که با یک الگوریتم تکاملی برای مسئله‌ی کلاسیک N-وزیر ارائه دادیم، در زمان نسبتاً مناسبی مسئله را برای Nهای معقول و مرسوم حل می‌کند و می‌توان دید که این بسیار بهتر از الگوریتم‌های دیگر از جمله عقبگرد است.

۴ منابع و مراجع

Evolutionary algorithms. (2020). In S. J. Russell, *Artificial Intelligence: A Modern Approach* (4 ed., pp. 115-119). Pearson. Retrieved from http://aima.cs.berkeley.edu/

Rajesh-Reddy1/Sample-CHESS-board. (n.d.). Retrieved from GitHub: https://github.com/Rajesh-Reddy1/Sample-CHESS-board

Ramin Saljoughinejad. (n.d.). Retrieved from https://www.youtube.com/: https://www.youtube.com/channel/UCgbfxzjrkxpQTimKlTHZSeg