دانشگاه صنعتی امیرکبیر
(پلی‌تکنیک تهران)
دانشکده ریاضی و علوم کامپیوتر

پروژه

علوم کامپیوتر

پیاده سازی بازی کاکورو

نگارش

مهدی عباسعلی پور

استاد راهنما

جناب آقای دکتر قطعی

استاد مشاور

جناب آقای یوسفی مهر

آذرماه ۱۴۰۲


بسم الله الرحمن الرحيم

چکیده

در این پروژه هدف بر این است که بازی کاکورو با استفاده از زبان برنامه نویسی پایتون پیاده سازی شود. بازی به این صورت می باشد که اعداد ۱ تا ۹ باید با توجه به قوانینی که در راهنمای پروژه آمده است در خانه ای جدول نوشته شوند . برای حل این بازی دو عامل پیاده سازی می شوند که عامل اول بدون هیچ گونه ابتکاری به دننبال جواب می کردد اما عامل دوم با استفاده از حدس هایی ابتدا خانه هایی خاص را پر می نماید .

واژه‌های کلیدی:

بازی کاکورو ، مسئله ارضای محدودیت، روش های فرا ابتکاری

فهرست مطالب

عنوان صفحه

فهرست تصاویر

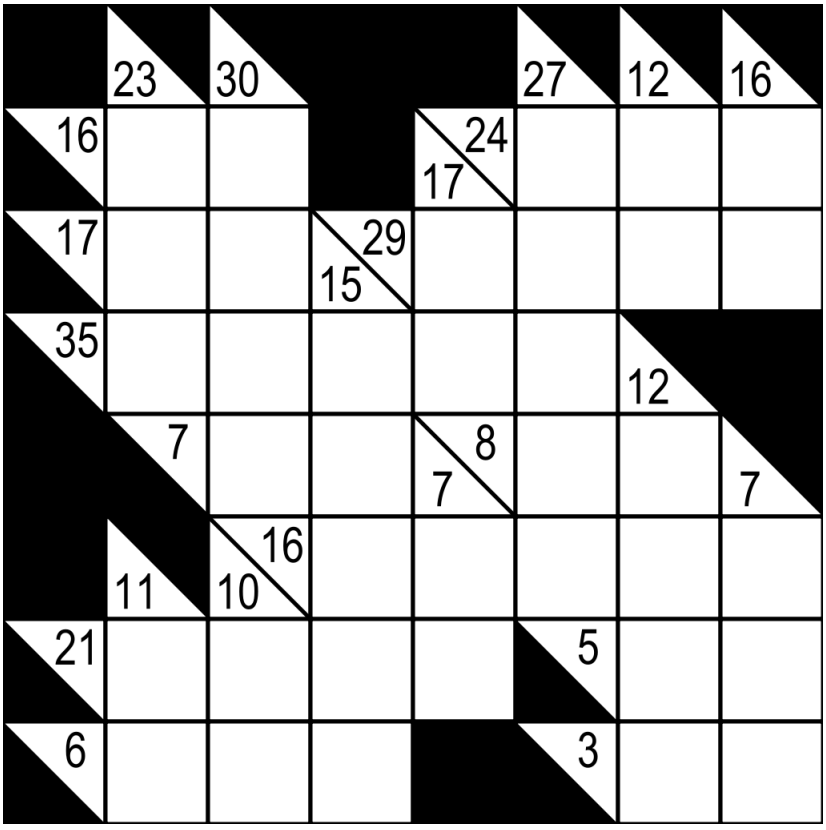
شکل صفحه

فصل اول

بازی کاکورو

۱-۱ مقدمه

کاکورو یک بازی معمایی می باشد. می توان اینگونه گفت که کاکورو شباهت زیادی با سودوکو دارد. همانطور که از اسم این معما بر می آید، کاکورو خاستگاهی ژاپنی دارد [۱] . جدول کاکورو از چند سطر و ستون که هرسطر یا ستون ممکن است چند بخش باشند تشکیل شده و می‌تواند ابعاد مختلفی داشته باشد. در سمت چپ یا بالای هر بخش عددی قید شده که نشانگر مجموع اعداد آن قسمت از سطر یا ستون است. سطرها و ستون ها توسط خانه های خاکستری رنگ از یکدیگر جدا شده اند. برای حل کاکورو باید از بین اعداد ۱ تا ۹، اعداد مناسب را برای هر سطر یا ستون طوری انتخاب نمایید که اولا مجموع اعداد آن سطر یا ستون با عدد نشان داده شده در ابتدای آن یکسان باشد و ثانیا در هیچ سطر یا ستونی عددی تکراری نباشد.



شکل ۱-۱: بازی kakuro
[۲]

۱-۲ طرح مسئله

در اسن مسئله بایستی با استفاده از الگوریتم پس گرد عاملی طراحی کنیم تا جدول را حل نماید و سپس این عامل را بهبود دهیم . درنهایت عامل ها را با یکدیگر مقایسه نماییم .

۱-۳ لینک گیت هاب کد

با مراجعه به https://github.com/mahdialipoo/AI_project4 می توانید کد مربوط به پیاده سازی این بازی را مشاهده نمایید .

فصل دوم

پیاده سازی بازی و عامل ها

۲-۱ محیط بازی

صفحه بازی به صورت آرایه ای جدولی از اعداد می باشد که خانه هایی که باصفر مشخص می شوند مجاز به دریافت مقدار می باشند و باقی خانه ها با ۱- پر می شوند . برای مشخص کردن هر بخش و حاصل جمع آن بخش از لیستی از قیود استفاده می شود .که مختصات خانه ی شروع و طول و حاصل جمع و عمودی بودن یا افقی بودن بخش در لیست نگهداری می شود .

۲-۲ مروری برتوابع اصلی پیاده سازی بازی

اگر دو تابع tab-solved و constrain-hope که جدول و قیود را به عنوان ورودی می پذیرند True شود جدول حل شده است .

توابع search-constrain-v و search-constrain-h مختصات یک خانه را می گیرند و به ترتیب قید های متناظر سطری و ستونی آن را باز می گردانند . اگر یکی از این قیود را نداشته باشد نیز مقدار True را باز می گردانند .

تابع search جدول و قید هابه همراه یک تابع selection را می گیرد و جدول را با جست و جو در فضای حالات به روش DFS حل می نماید . تابع selection در هر مرحله مشخص می نماید که کدام خانه مقدار دهی شود . با تغییر این تابع می توان عاملی را که کل فضای حالات را می گردد تبدیل به عاملی کرد که با روشی ابتکاری به جست و جو می پردازد و جست و جو سمت و سو می دهد .

۲-۳ عامل های بازی

در بخش پیشین این نکته ذکر شد که با تغییر تابع selection که به عنوان ورودی تابع search است می توان عامل را هوشمند کرد .

۲-۳-۱ عامل اولیه

در ابتدا انتخاب برمبنای این بود که خانه ها به ترتیب از مختصات مبدا مقدار دهی شوند . بهترتیب مقادیر ۱ تا۹ را بپذیرند و امید این که به جواب می رسد در هر مرحله بررسی شود که زمان حل بسیار طولانی می شد .

پس از این آزمون اولیه انتخاب خانه ها به صورت تصادفی انجام می شد که زمان حل بهبود یافت .

۲-۳-۲ عامل هوشمند

با استفاده از استراتژی انتخاب محدود ترین متغیر به مقدار دهی متغیر ها پرداخته شد. این نوع انتخاب در تابع select-cell-intelligent پیاده سازی شده است . در این تابع ابتدا متغیر هایی مقدار دهی می شوند که مجموع اعداد مربوط به قید آن ها کمترین باشد و به این ترتیب سرعت حل مسئله کاکورو افزایش می یابد .

فصل سوم

جمع بندی و نتیجه گیری

با استفاده از الگوریتم back tracking سعی شد تا بازی کاکورو حل شود وباتوجه به طولانی بودن زمان حل به این روش در ادامه سعی شد به روش های فرا ابتکاری مسلح شود . از مهم ترین چالش ها پس از پیاده سازی روش انتخاب بهترین خانه بود که سعی شد با معیار های متفاوتی انجام پذیرد و در نهایت با استفاده از LCV [1] سعی شد این بهبود حاصل شود . اما به علت زیاد بودن محاسبات حل یک جدول واقعی ۸ در ۸ توسط هیچ کدام از عامل ها انجام نشد . در مقابل حل جدول آکاکورو آسان تر ۵ در ۵ توسط این عامل ها آزمایش شد . عامل اولیه برای حل جدول در حدود ۱۵ ثانیه زمان گرفت و عامل هوشمند در چند دهم ثانیه جدول را حل کرد . عامل رندوم هم بعضی موارد زمان بسیار بالایی نیاز داشت و در برخی موارد در حدود زمان ۲۰ثانیه جدول را حل می کرد . برای جدول ۶ در ۶ که سطح سختی آسان داشت عامل اولیه در دو دقیقه جواب را پیدا کرد و عامل هوشمند درحدود۵ ثانیه جواب را یافت . عامل رندوم هو عملکرد زمانی بسیار ضعیفی داشت . جدول ۶ در ۶ مذبور در پوشه veryeasy قرار دارد .

```
tab1=tab.copy()
search(tab1,constrain,selection=select_cell)
✓ 2m 11.2s

True

tab1
✓ 0.0s

array([[-1, -1, -1, -1, -1, -1],
       [-1, -1,  3,  1,  8, -1],
       [-1,  1,  4,  2,  3, -1],
       [-1,  2,  1, -1,  1, -1],
       [-1, -1,  2,  1, -1, -1],
       [-1, -1, -1,  2,  1, -1]])
```

شکل ۳-۱: زمان اجرای عامل اولیه

```
[39] tab3=tab.copy()
     search(tab3,constrain,selection=select_cell_intelligent)
     ✓ 4.3s

...  True

[40] tab3
     ✓ 0.0s

...  array([[-1, -1, -1, -1, -1, -1],
            [-1, -1,  3,  1,  8, -1],
            [-1,  3,  1,  2,  4, -1],
            [-1,  1,  2, -1,  1, -1],
            [-1, -1,  4,  1, -1, -1],
            [-1, -1, -1,  2,  1, -1]])
```

شکل ۳-۲: زمان اجرای عامل هوشمند

[1] Least Constraining Value

مراجع

[1] jadvalyab. how to play kakuro. , 2023.

[2] wikioedia. Kakuro. , 2023.