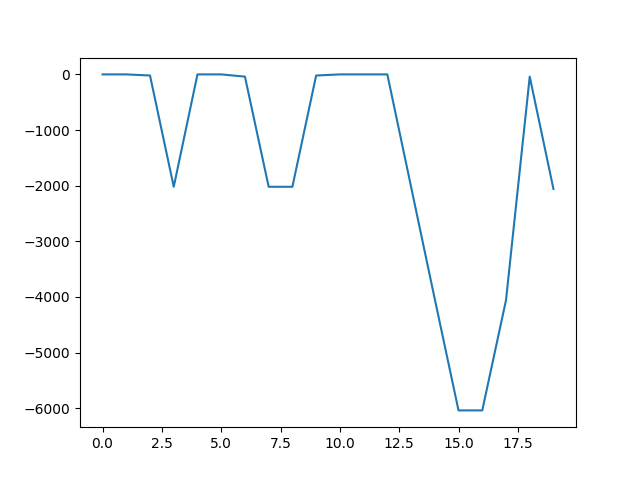
مهدی سلمانی صالح­آبادی 98105824

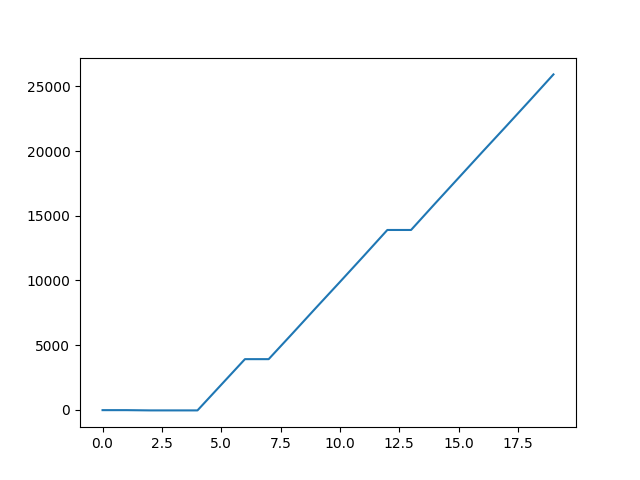
حالت اول) نفر اول رندوم، نفر دوم رندوم (نفر دوم برد)

Result: -1 Client1\_Delay: 0.0 Client2\_Delay: 0.001

چون هر دو نفر رندم بازی کردند زمانی برای تصمیم گیری استفاده نکردند. (دیلی جفتشون تقریبا صفر است) و نتیجه هم تصادفی است یعنی اگر یک بار دیگر بازی انجام می­شد ممکن بود نفر اول ببرد.

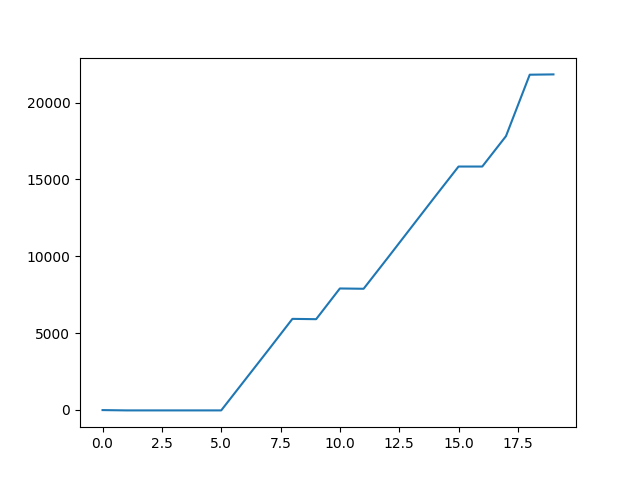
حالت دوم) نفر اول رندوم، نفر دوم minmax با عمق یک (نفر اول برد)

Result: 1 Client1\_Delay: 0.0 Client2\_Delay: 0.005

اینکه نفر دوم به دلیل بررسی بهترین انتخاب خود زمان صرف کرده واضح است. ولی چرا اینقدر بد شکست خورد؟ چون عمق بررسی این فرد کم است، نمی­تواند حالاتی را که در آینده برای او در دو حرکت 1000 امتیاز میاورند را در ابتدا تشخیص دهد و برای همین دید محدود سعی می­کند بیشتر 10 امتیاز 10 امتیاز، به امتیاز خود بیفزاید. (اگر جدول بازی را نگاه کنید کل مهره­ها را در یک ستون قرار می­دهد زیرا با بینش عمق یکی گذاشتن مهره در ستون بقلی (که بعدا می­تواند منجر به 1000 امتیاز شود) توجیه ندارد.)

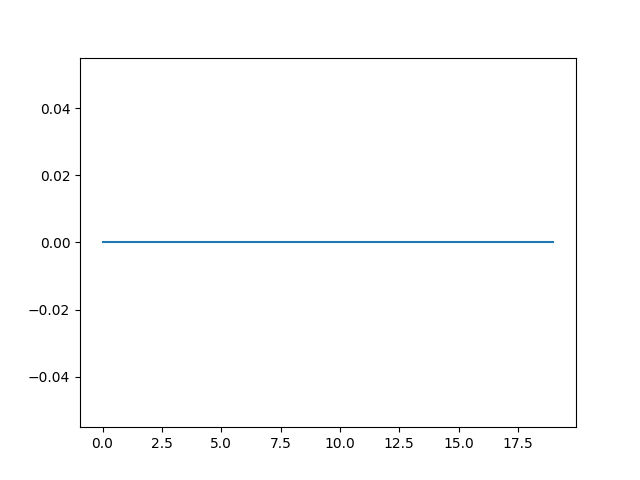
حالت سوم) نفر اول رندوم، نفر دوم pruning با عمق یک (نفر اول می­برد)

Result: 1 Client1\_Delay: 0.001 Client2\_Delay: 0.006

دلیل برد نفر اول در قسمت قبل بیان شد. از نظر سرعت اما انتظار pruning سریعتر باشد. (از minmaxاصلی) ولی اینگون نشد. چرا؟ چون در عمق یک اصلا تاثیری ندارد و باعث کهش تعداد حالات بررسی نمی­شود.

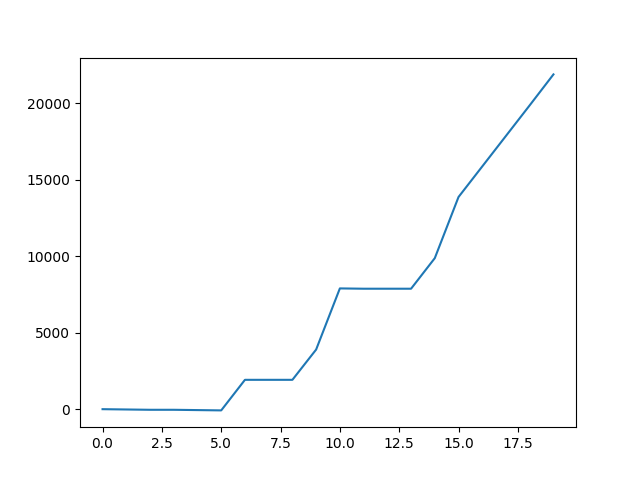
حالت چهارم) نفر اول orginal\_minmax و نفر دوم pruning هر دو با عمق یک (بازی مساوی شد)

Result: 0 Client1\_Delay: 0.005 Client2\_Delay: 0.005

اینکه از نظر زمانی چرا بهبود نداریم در قسمت سوم بیان شد. اما چرا بازی مساوی شد. چون عمق جواب کم است هر دو نفر تمام مهره­هایشان را در یک ستون قرار دادند و نتیجه اینگونه شد. ( در واقع متقارن بازی کردند.)

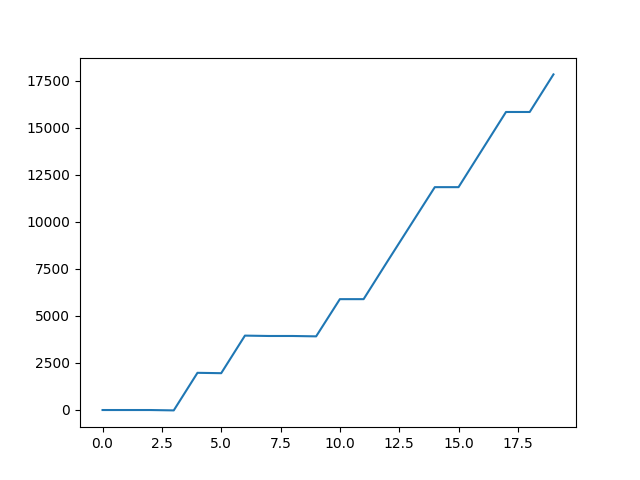
حالت پنجم) نفر اول رندوم، نفر دوم minmax با عمق دو (نفر اول برد)

Result: 1 Client1\_Delay: 0.001 Client2\_Delay: 0.046

دقیقا مانند حالت دوم است. البته به دلیل عمق بیشتر از حالت دوم زمان تاخیر نفر دوم بیشتر شده­است ولی دوباره چون نمی­تواند با عمق دو امتیاز 1000 را تشخیص دهد (از ابتدا) تمام مهره­ها را روی یک ستون قرار می­دهد. (در واقع عمق دوم فقط باعث می­شود که حرکت حریف را حدس بزند و باعث نمی­شود که خودش بتواند به گونه­ای مهره بگذازد که امتیاز 1000 را بگیرد.)

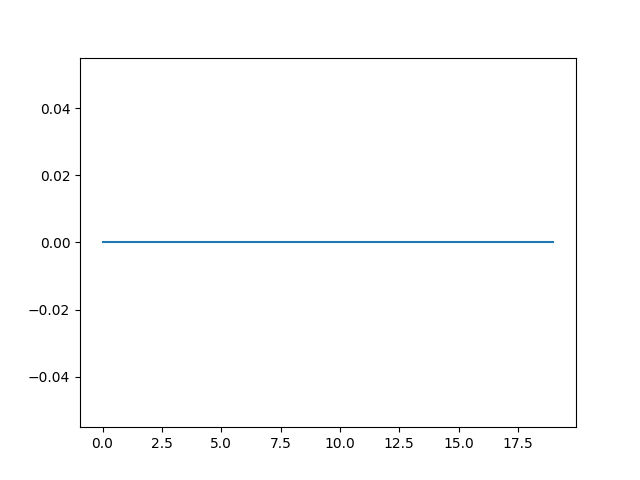
حالت ششم) نفر اول رندوم، نفر دوم pruning با عمق دو (نفر اول می­برد)

Result: 1 Client1\_Delay: 0.001 Client2\_Delay: 0.08

دلیل اینکه نفر اول می­برد همان دلیل حالت قبلی است. اما چرا از نظر زمانی دوباره بهبود نداریم؟ چون دوباره روش pruning تا عمق دو روش کارایی نیست و بیشتر باعث اتلاف وقت می­شود.

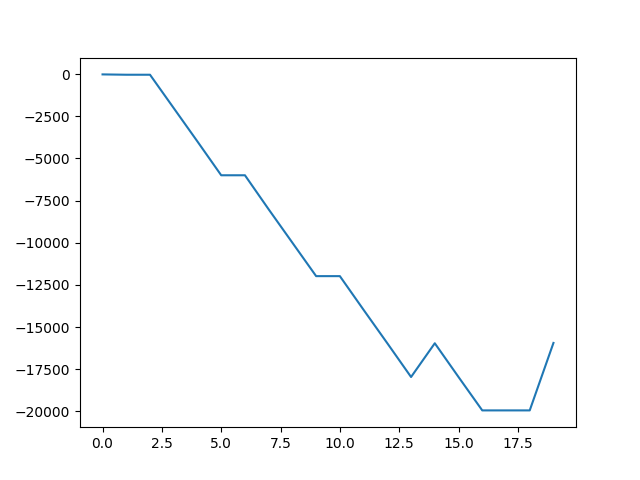
حالت هفتم) نفر اول orginal\_minmax و نفر دوم pruning هر دو با عمق دو (بازی مساوی شد)

Result: 0 Client1\_Delay: 0.172 Client2\_Delay: 0.174

دقیقا مانند حالت چهارم است. هم دلیل مساوی­شدن و هم دلیل بهبود نیافتن زمانی الگوریتم pruning

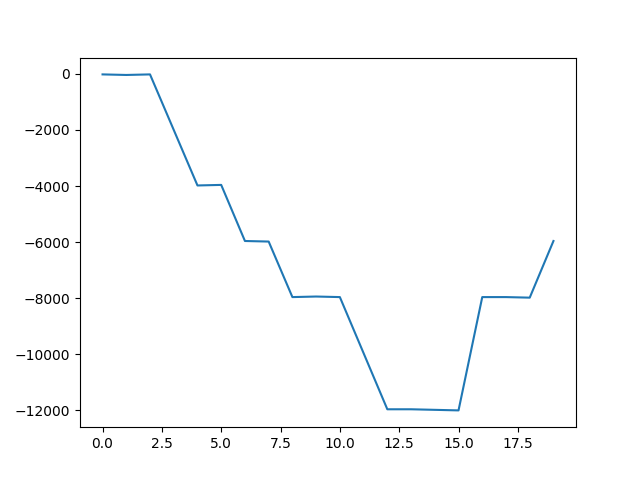
حالت هشتم) نفر اول رندوم، نفر دوم minmax با عمق سه (نفر دوم برد)

Result: -1 Client1\_Delay: 0.001 Client2\_Delay: 1.056

از این­جا به بعد نفر دوم توانایی روش گرفتن 1000 امتیاز را دارد و دیگر در یک ستون مهره نمی­گذارد. برای همین با اختلاف نفر اول که رندوم بازی می­کرد را شکست می­دهد.

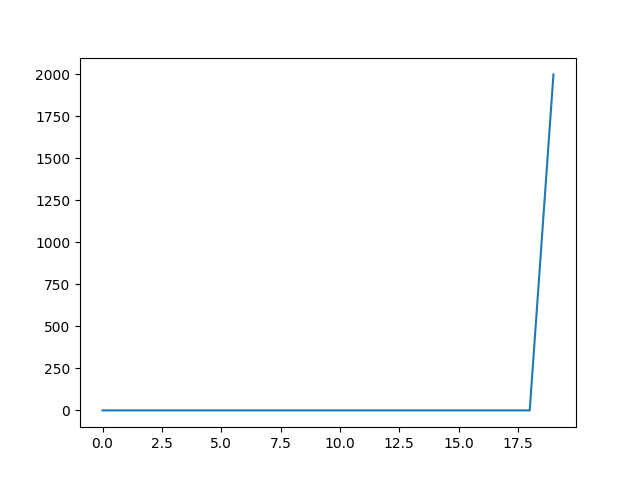
حالت نهم) نفر اول رندوم، نفر دوم pruning با عمق سه (نفر دوم می­برد)

Result: -1 Client1\_Delay: 0.0 Client2\_Delay: 0.414

دلیل برد نفر دوم مانند حالت قبلی است. اما دقت کنید بالخره ما تاثیر pruning را در الگوریتم مشاهده کردیم که باعث افزایش سرعت minmax تا عمق 3 شد و از نظر زمانی نصف original minmax زمان برد.

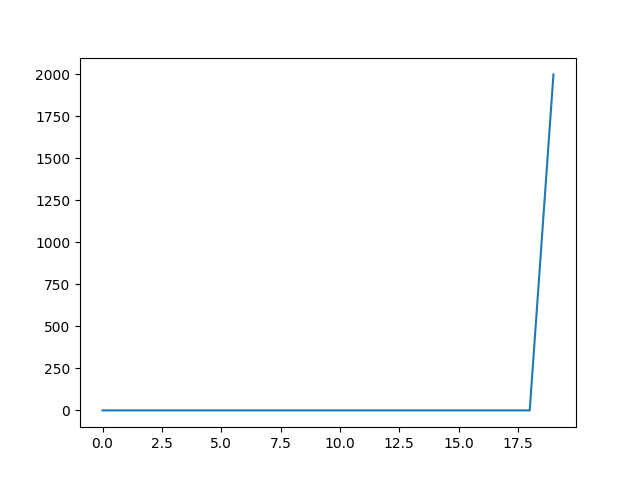
حالت دهم) نفر اول orginal\_minmax و نفر دوم pruning هر دو با عمق سه (نفر اول برد)

Result: 1 Client1\_Delay: 1.811 Client2\_Delay: 1.493

از نظر زمانی همانطور که انتظار می­رفت نفر دوم سریعتر است. اما چرا نفر اول برد و بازی مساوی نشد؟ چونکه دو نفر در این حالت مهره­هایشان را در یک ستون نمی­گذارند و برای همین درست است که متقارن بازی می­کنند ولی جایی می­رسد که هر دو یک خانه را می­خواهند و خب در این حالت نفر اول که بازی را شروع کرد دست پیش را دارد.

حالت یازدهم) نفر اول pruning با عمق چهار و نفر دوم pruning با عمق سه (نفر اول برد)

Result: 1 Client1\_Delay: 23.199 Client2\_Delay: 1.622

 در این حالت نیز مانند حالت قبلی تا جایی از باز دو طرف متقارن بازی می­کنند ولی خانه­ای در صفحه به وجود می­آید که جفتشان آن خانه را می­خواهند و ابتکار دست نفر اول است. (دقت کنید افزایش عمق کمی تاثیرگذار است ولی دلیل اصلی برد نفر اول، نفر اول بودن است)

از نظر زمانی هم دقت کنید که اگر minmax با عمق 4 انجام می­دادیم میزان تاخیر خیلی خیلی زیاد می­شد در حدی که احتمالا برنامه را terminate می­کردیم.