Uma Game Battle Platform baseada em Web-API para Educação em Inteligência Artificial

1º Xiaofei Han Beijing
Laboratório Chave de
Telecomunicações Inteligentes.
Software e Multimídia
Universidade de Correios e
Telecomunicações de Pequim.
Pequim 100876, China
xfhan@bupt.edu.cn

2º Junwei Zou Escola
de Engenharia Eletrônica
Universidade de
Correios e Telecomunicações
de Pequim.
Pequim 100876, China
buptzjw@bupt.edu.cn

3º Laboratório-
Chave de Telecomunicações
Inteligentes de Wei Ren Beijing.
Software e Multimídia
Universidade de Correios e
Telecomunicações de Pequim.
Pequim 100876, China
rw478463266@gmail.com

4º Laboratório-
Chave Yan Sun Beijing de
Telecomunicações Inteligentes.
Software e Multimídia
Universidade de Correios e
Telecomunicações de Pequim.
Pequim 100876, China
sunyan@bupt.edu.cn

Resumo—O potencial dos jogos de computador como ferramenta para pesquisa e educação em IA continua a florescer. A plataforma de batalha de jogos fornece um lugar para os alunos aprenderem teoria de inteligência artificial e praticarem algoritmos de inteligência artificial.
No entanto, as plataformas existentes têm um desempenho ruim em termos de alta simultaneidade, bot de depuração e suporte à linguagem Scratch. Para resolver esses problemas, este artigo apresenta primeiro um mecanismo de batalha de API web usando o protocolo Websocket. Aumenta o número de correspondências simultâneas nos servidores. Em seguida, fornecemos uma ferramenta de batalha offline para facilitar a depuração e o teste de bots para os usuários. Além disso, o ambiente especial é introduzido para suportar bots de programação no Scratch. Por fim, realizamos o experimento de comparação entre o mecanismo proposto e os métodos tradicionais de batalha. Os resultados mostram que ele suporta maior simultaneidade e reduz o uso da CPU em 40%.

Termos de indexação — batalha de jogos, educação em IA, Scratch

I. INTRODUÇÃO

A Inteligência Artificial é um campo em rápido desenvolvimento, atraindo um número crescente de pesquisadores e aprendizes nos últimos anos. É difícil e monótono para a maioria dos adolescentes aprender sobre estruturas de dados e algoritmos de inteligência artificial. Atualmente, as plataformas de batalha de jogos aumentam efetivamente a motivação de código dos adolescentes, combinando jogos nos cursos de programação e inteligência artificial [1].
Os adolescentes podem desenvolver o código do bot com algoritmos inteligentes de acordo com a regra do jogo e competir contra outros bots.

As plataformas de batalha de jogos [2] são geralmente divididas em plataformas de batalha offline e online. No trabalho existente, a plataforma offline está executando o bot no computador do usuário. É difícil para os usuários comparar seus bots com outros. A plataforma online está sempre executando o bot no servidor, portanto, o número de correspondências simultâneas é limitado. Além disso, os servidores sempre executam correspondências inválidas devido ao formato de saída ilegal dos bots, o que pode causar um desempenho ruim dos servidores.
Além disso, as plataformas existentes são principalmente para estudantes do ensino médio e universitário e não suportam a popular programação Scratch [3].

Para resolver os problemas acima, este artigo propõe uma nova plataforma de batalha de jogos que combina módulos offline e on-line. módulos de linha. Além disso, ele suporta várias linguagens de programação e uma variedade de jogos interessantes baseados em turnos. Especificamente, primeiro projetamos o mecanismo de batalha da API web [4] usando o protocolo Websocket para melhorar o número de correspondências simultâneas. Quando o servidor está executando uma partida, os bots estão sendo executados no computador do usuário em vez do servidor. Assim, ele pode suportar mais partidas ao mesmo tempo. Em segundo lugar, desenvolvemos uma ferramenta de batalha offline para facilitar a depuração local dos bots. Reduz correspondências inválidas incorridas por formato de saída ilegal na plataforma. Terceiro, para fazer com que os jovens alunos experimentem a diversão do aprendizado de programação, apresentamos o ambiente especial para suportar bots de programação no Sc
As crianças podem desenvolver bots arrastando blocos de construção para competir com outras pessoas usando outros idiomas. Por fim, realizamos um experimento de comparação entre o mecanismo proposto e os métodos tradicionais de batalha. Os resultados experimentais comprovam que o mecanismo proposto possui alta simultaneidade e baixa carga do servidor.

II. TRABALHO RELACIONADO

Existem muitas formas de plataformas de batalha de jogos. Por exemplo, a Associação Internacional de Jogos de Computador (ICGA) é responsável por sediar o Campeonato Internacional de Jogos Olímpicos de Computador, que se compromete a fornecer uma plataforma internacional para atividades de jogos de computador. O jogo inclui 10 itens como Amazon Chess, Gomoku e Go [5]. Como as correspondências são feitas apenas periodicamente, os usuários geralmente só podem programar e testar os bots offline. É difícil para os usuários aprenderem uns com os outros. A Alloyteam desenvolve um jogo de programação online chamado CodeTank, que é uma plataforma competitiva de jogos de programação que permite aos jogadores aprender e melhorar a programação Javascript e a pesquisa de IA [6]. No entanto, a plataforma só pode suportar JavaScript e o jogo único. Tanto Riddles.io [7] quanto Botzone [8] suportam vários idiomas e uma variedade de jogos. Os bots no Riddles.io estão sendo executados no servidor, portanto, várias correspondências simultâneas causarão alto uso da CPU do servidor. Além disso, os usuários não podem ver a partida até que ela termine de julgar. Botzone é desenvolvido e mantido pelo laboratório de IA da Universidade de Pequim.

978-1-7281-3129-0/20/$31,00 ©2020 IEEE

Embora suporte o mecanismo de batalha da API, é baseado no protocolo HTTP. Você precisa verificar se há novas mensagens por sondagem, o que é um desperdício de recursos. E é complicado transformar o bot em um bot com o mecanismo web-API. Além disso, nenhuma dessas plataformas suporta a programação Scratch.

Neste artigo, propomos primeiro um mecanismo de batalha de API web usando Websocket. Ele pode resolver o problema de baixas correspondências simultâneas. Em segundo lugar, desenvolvemos uma ferramenta de batalha offline que é conveniente para os usuários testarem e depurarem bots localmente. Em terceiro lugar, apresentamos o ambiente especial para dar suporte aos bots do Scratch para a batalha.

III. VISÃO GERAL DO SISTEMA

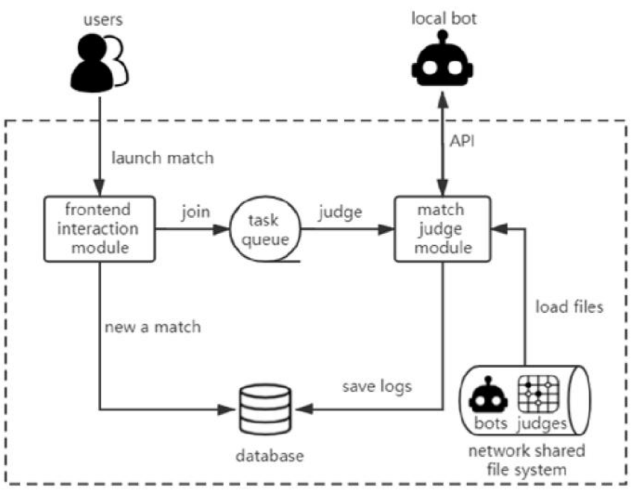
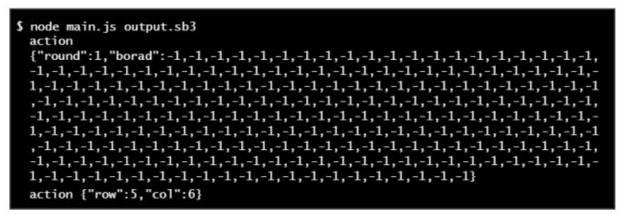


Fig. 1. Partes principais da plataforma.

Conforme mostrado na Figura 1, o sistema proposto é composto principalmente por módulo de interação frontend, módulo match judge e módulo de armazenamento de dados. No módulo de interação de front-end, os usuários podem fazer upload de seus programas como bots, iniciar partidas, assistir a partidas ao vivo e reproduzir partidas. O módulo de juiz de jogo atua como um papel central na plataforma. Ele é responsável por julgar a partida no ambiente sandbox, determinar o resultado da partida e enviar o log em tempo real da partida para o módulo frontend. O módulo de armazenamento de dados inclui um sistema de arquivos compartilhado em rede e um banco de dados. O sistema de arquivos compartilhados em rede é responsável por armazenar o código dos juízes e bots do jogo, e o banco de dados é responsável por armazenar os registros das partidas e os dados da plataforma. A nova partida entra na fila de tarefas e, em seguida, o módulo de juiz da partida realiza a partida. Ele é executado em um ambiente sandbox, que pode isolar o servidor do ambiente da máquina do juiz. Assim, ele pode impedir que bots maliciosos ataquem o servidor. O módulo de juiz de partida executa o juiz de jogo e os bots necessários e registra o resultado do jogo. Durante a partida, os usuários podem assistir à partida ao vivo no frontend. Após o término da partida, o resultado da partida é salvo no banco de dados para reprodução.

Há dois grandes destaques para a plataforma: maiores correspondências simultâneas e suporte à linguagem Scratch. Primeiro, a plataforma de batalha do jogo usa o mecanismo de API da Web para batalhar e fornece carregador de bot para experiência do usuário. O carregador de bot carrega e executa o bot localmente e transmite informações em tempo real com o servidor usando o protocolo Websocket. Reduz bastante a pressão sobre o servidor. Além disso, os usuários podem usar a ferramenta de batalha offline para testar e executar bots, o que pode reduzir o servidor a realizar algumas correspondências inválidas e otimizar o desempenho do servidor indiretamente. Em segundo lugar, a plataforma suporta uma variedade de linguagens de programação convencionais, incluindo Python, C/C++, Java, etc. Além disso, também suportamos a linguagem Scratch para programação de bots para adolescentes. Especificamente, introduzimos um ambiente especial baseado no Scratch 3.0, que pode executar o arquivo Scratch .sb3 no terminal. O uso desse ambiente possibilita a interação de dados com o sb3 no terminal. Conforme mostrado na Figura 2, o ambiente está executando o bot Scratch do jogo Gobang.



```
$ node main.js output.sb3
  action
  {"round":1,"borad":-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
  -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
  1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
  ,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
  -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
  1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1
  ,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,
  -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-
  1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1}
  action {"row":5,"col":6}
```

Fig. 2. Execute o bot do Scratch no terminal.

4. MÉTODOS PARA ALTA CONCORRÊNCIA

A. Mecanismo de API da Web

A arquitetura do mecanismo web-API é mostrada em Figura 3, e é desenvolvido com base no mecanismo de batalha tradicional. Judge é o código de uma regra do jogo que é usado para aceitar as saídas dos bots e dar instruções aos bots. O servidor Websocket é responsável por enviar instruções para o bot local e enviar as informações recebidas para julgamento.

O middleware de correspondência é o núcleo do mecanismo de API da web. Ele desempenha o papel de ponte de comunicação entre o juiz e o servidor Websocket. O algoritmo principal do middleware de correspondência é mostrado no Algoritmo 1. Todo o processo pode ser dividido em três etapas: execução do programa (linha 2-6), escuta da mensagem (linha 7-21) e encaminhamento da mensagem (linha 23-26).).

Na primeira etapa, o algoritmo carrega o arquivo de configuração de correspondência e, em seguida, executa o juiz e o servidor Websocket.

Na etapa dois, o algoritmo ouve a mensagem do juiz e analisa o formato da mensagem que é mostrado na Tabela 1. O processamento correspondente é realizado de acordo com diferentes formatos de mensagem.

Na etapa três, o algoritmo encaminha a mensagem e registra no registro de partida.

A fim de tornar mais fácil para os usuários usar a API da web mecanismo para a batalha, nós encapsulamos a parte de comunicação

TABELA I
FORMATO DE MENSAGEM ENTRE JUIZ E BOT LOCAL.

	Instrução	Função
Juiz para bot	valor de definições configuração [tipo]	Inicialize a configuração do jogo
	perguntar [botId] ação [tipo] [Tempo]	Enviar comando para o bot
bot para ação de juiz [actionDetail]		Informação de decisão de saída

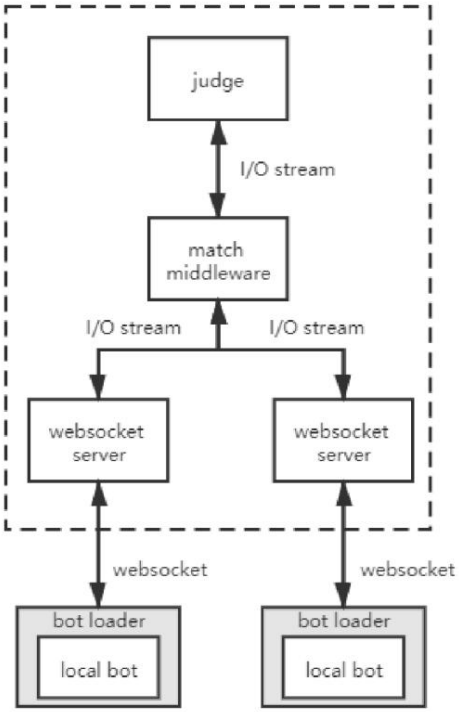


Fig. 3. Arquitetura do mecanismo web-API.

do Websocket no programa de PC do carregador de bot, conforme mostrado em Figura 4. O algoritmo central é mostrado no Algoritmo 2. O bot loader é responsável por encaminhar as informações do servidor para o bot local e encaminhando a saída dos bots para o servidor. Os usuários podem usar o mecanismo de API da web para batalhar sem adicionando códigos de comunicação adicionais manualmente no bot.

O mecanismo web-API pode reduzir a pressão do servidor. Além disso, é possível fazer pleno uso dos recursos locais CPUs para encurtar o tempo de execução do bot. Neste papel, o mecanismo estende a arquitetura do mecha tradicional nism na forma de um plugin. No mecanismo tradicional, o servidor executa bots em vez de códigos de servidor Websocket. este design estendido permite que a plataforma suporte a partida entre o bot de execução local com mecanismo de API da Web e bots com o mecanismo tradicional.

Algoritmo 1 Corresponder Middleware

Entrada: configuração de correspondência
Saída: resultado da partida

```
1: função JUIZ(configuração)
2: Juiz ÿ CRIAR(conf
3:     ig, gameJudgeP ath)
4:     para i = 0 ÿ botP aths.length do
5:         bot[i] ÿSPAWN(config.botP ath[i])
6:     fim para
7:     enquanto a verdade faz
8:         mensagem ÿ esperar mensagem do Juiz
9:         if message[”tipo”] == ”configuração” then
10:            para i = 0 ÿ bot.length do
11:                SENDMESSAGE(bot[i], mensagem)
12:            fim para
13:        senão se mensagem[”tipo”] == ”perguntar” então
14:            id ÿ mensagem[”id”]
15:            SENDMESSAGE(bot[id], mensagem)
16:            mensagem ÿ espera mensagem do bot
17:            ENVIO MENSAGEM(Juiz, mensagem)
18:        senão
19:            retornar resultado
20:        fim se
21:    terminar enquanto
22: função final
23: função ENVIO MENSAGEM(processo, mensagem)
24:    enviar mensagem para stdin do processo
25:    atualizar resultado global
26: função final
```

Carregador de bot do **algoritmo 2**

Entrada: URL com privateKey e botCommand
Saída: Vazio

```
1: função RUN(url, privateKey, botCommand)
2: ws ÿ CREATEWEBSOCKET(url, privateKey)
3:     bot ÿ SPAWN(botCommand)
4:     ouvir mensagem de ws
5:         SENDMESSAGETOPROCESS(bot, mensagem)
6:     ouvir mensagem do bot
7:         ws enviar mensagem para o servidor
8: função final
```

B. ferramenta de batalha offline

Nós fornecemos uma ferramenta de batalha offline para desktop de PC. Como mostrado na Figura 5, o módulo de juiz de partida é o mesmo que o módulo online plataforma de batalha, que pode restaurar o ambiente do partida on-line. Os alunos usam esta ferramenta para testar e desafiar o bot localmente. Pode reduzir correspondências inválidas incorridas por formato de saída no servidor.

A ferramenta de batalha offline pode não apenas facilitar a depuração, mas também permite que usuários offline se divirtam com a luta de código.

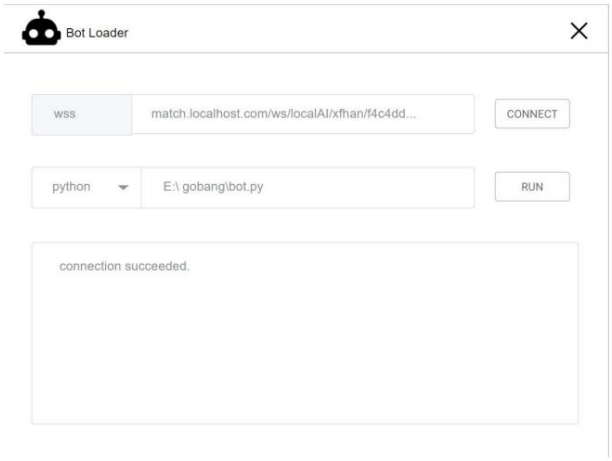


Fig. 4. O carregador de bot.

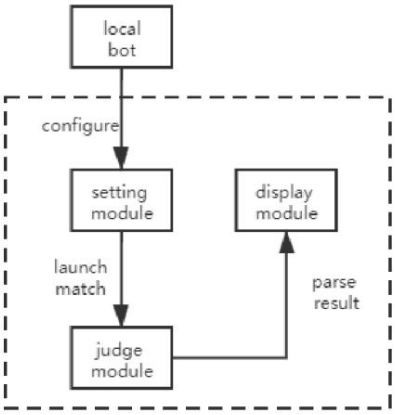


Fig. 5. Estrutura da ferramenta de batalha offline

V. EXPERIMENTO

A. Configuração do experimento

Os experimentos são conduzidos na área de trabalho do Windows 10 com CPU de 4 núcleos, 16 GB de RAM e executando o navegador Chrome versão 77.0.3865.120. Os clientes e o servidor usam a mesma configuração do computador no experimento. Além disso, usamos o Celery como a fila de tarefas distribuídas.

Para avaliar o mecanismo de web-API, comparamos com o mecanismo tradicional chamado bot de nuvem no jogo Gobang. 10 testadores usaram o mesmo bot e dois em grupos jogam uns contra os outros. Os primeiros testadores usam o mecanismo de bot de nuvem para iniciar a partida ao mesmo tempo. Após o término de todas as correspondências, o testador executa o carregador de bot para executar o bot local em seu computador e, em seguida, usa o mecanismo de API da Web para iniciar a correspondência ao mesmo tempo.

Para fazer uma comparação justa, precisamos garantir que os ambientes de computação de todos os bots sejam consistentes. No mecanismo de bot de nuvem, cada núcleo de CPU do servidor executa uma correspondência. Assim, o modo de multiprocessamento de aipo é usado para executar o mecanismo. No mecanismo web-API, os bots são executados no computador do usuário, o que garante que o ambiente computacional não seja afetado pela CPU do servidor. Portanto, o mecanismo de API da Web usa o modo multithreading no aipo.

B. Aprendizado de amostra

O jogo de Gobang é um jogo simples e conhecido jogado em um tabuleiro de 15 * 15. Os jogadores alternam turnos colocando uma pedra de sua cor em uma interseção vazia. O vencedor é o primeiro jogador a formar uma cadeia ininterrupta de cinco pedras na horizontal, vertical ou diagonal [9]. Fornecemos código de juiz sobre a regra de Gobang, um player de exibição de jogos e amostras de bot em vários idiomas. As amostras de bot fornecem o código básico de comunicação de dados e a lógica do algoritmo para os desenvolvedores de bot. Além disso, desenvolvemos alguns blocos de construção para as amostras Scratch do Gobang. É conveniente que o usuário se concentre no desenvolvimento do algoritmo principal. A amostra de bot do Scratch 3.0 para o jogo Gobang é mostrada na Figura 6.

O jogador 1 desenvolve um bot com base na amostra do Scratch fornecida

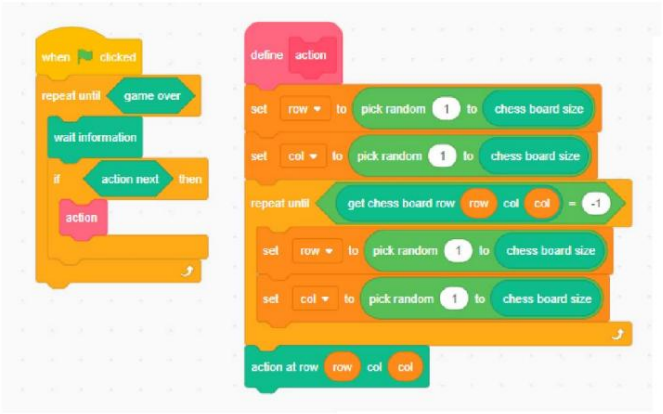


Fig. 6. Amostra do Scratch bot.

pela plataforma, e o Jogador 2 desenvolve um bot baseado no exemplo do Python. Após o desenvolvimento, os dois jogadores usam o carregador de bots para inserir a chave privada e o comando para executar seus bots e entrar na mesma sala para a batalha. Em seguida, eles podem assistir à partida em tempo real no player HTML5.

C. Análise de desempenho

TABELA II
RESULTADOS ENTRE OS DOIS MECANISMOS.

Característica	web-API de bot na nuvem	
tempo total para concluir todas as	145s	97s
partidas uso médio da CPU	66,3%	21,0%
tempo médio da rodada tempo	1015 ms	1067 ms
médio da partida	71,1s	74,0 segundos

O resultado experimental é mostrado na Tabela 2, que comprova que o mecanismo de web-API tem maior simultaneidade do que o método de bot de nuvem. Quando várias correspondências são iniciadas ao mesmo tempo, o servidor não inicia a execução de uma nova tarefa de correspondência até que haja uma CPU ociosa. No entanto, na API da Web

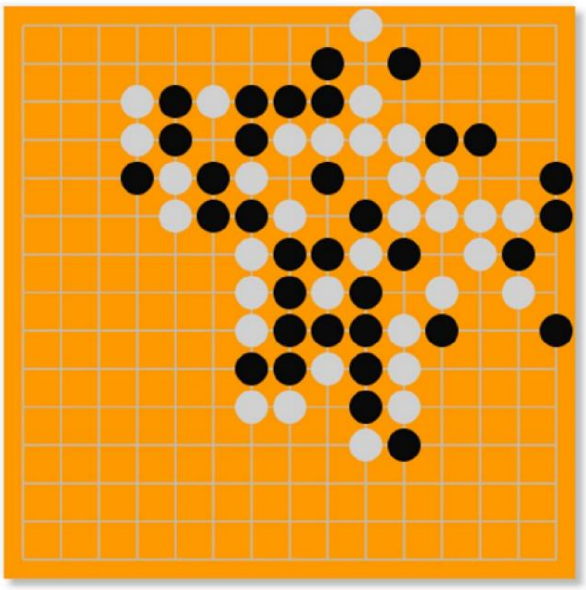


Fig. 7. Uma situação do jogo Gobang.

mecanismo em multithreading, as tarefas de correspondência podem ser executadas simultaneamente. Devido à sobrecarga de comunicação entre o bot local e o servidor, o tempo médio de rodada dos bots e o tempo médio de correspondência no mecanismo da API da Web são mais do que o método de bot de nuvem.

D. Comparação de plataforma

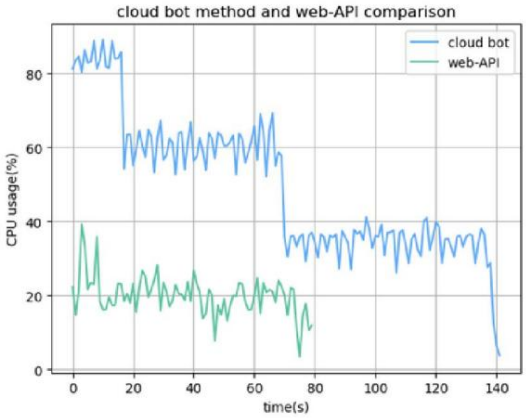


Fig. 8. Comparação do resultado entre os dois mecanismos.

TABELA III
COMPARAÇÃO COM OUTRAS PLATAFORMAS.

Característica	Riddles.io	Botzone	Nossa plataforma
Ferramentas de correspondência local	Sim	Não	Sim
Protocolo da API	/	Http	Websocket
Carregador de bot local	/	Não	Sim
Suporte ao bot do Scratch	Não	Não	Sim

A comparação entre nossa plataforma e outras é mostrada na Tabela 3, o que comprova que há grandes vantagens em nosso plataforma. Suporta mecanismo web-API usando Websocket protocolo e linguagem Scratch. Além disso, desenvolvemos um carregador de bot para fazer com que os usuários batalhem facilmente com a proposta mecanismo. No geral, esta plataforma oferece alta simultaneidade e é simples de usar.

VI. DESCOBERTAS

Em resumo, propomos uma plataforma de batalha de jogos para IA educação que suporta batalha offline e batalha online de alta simultaneidade. Além disso, ele suporta uma variedade de programação idiomas incluindo Scratch. Especificamente, primeiro projetamos o estrutura do mecanismo web-API que pode melhorar simultaneidade na execução das partidas. Em segundo lugar, desenvolvemos a ferramenta de batalha offline para ajudar os usuários na depuração e teste bots localmente. Terceiro, introduzimos um ambiente especial para apoie o Scratch para desenvolver bots para adolescentes. Por fim, realizamos uma experimento comparativo entre o mecanismo web-API e o mecanismo de bot de nuvem. Os resultados mostram que a web-API mecanismo é mais concorrente e ocupa menos recursos no servidor. A limitação do mecanismo web-API é útil apenas para as partidas disputadas por ambos os jogadores usando o carregador de bot local.

No futuro, iremos propor um mecanismo melhor para reduzir carga do servidor, não importa como os jogadores participem. Nós também suporte ao modo de classificação, no qual os alunos podem ver sua classificação na tabela de classificação.

RECONHECIMENTO

Este trabalho é parcialmente apoiado pela Fundação Nacional de Ciências Naturais da China sob a concessão 61877005, 61672109 e 61772085.

REFERÊNCIAS

[1] M. Papastergiou, “Digital game-based learning in high school computer ensino de ciências: impacto na eficácia educacional e no aluno motivação”, Computers & Education, vol. 52, nº. 1, pp. 0–12.
[2] F. Lu, K. Yamamoto, LH Nomura, S. Mizuno, Y. Lee e R. Tha ganharam, “Plataforma de competição de inteligência artificial de jogos de luta”, em 2013 IEEE 2ª Conferência Global de Eletrônicos de Consumo (GCCE). IEEE, 2013, pp. 320-323.
[3] M. Resnick, J. Maloney, A. Monroy-Hernandez, N. Rusk e YB Kafai, “Scratch: Programming for all,” Communications of the Acm, voar. 52, nº. 11, pág. 60-67, 2009.
[4] H. Zhou, H. Zhang, Y. Zhou, X. Wang e W. Li, “Botzone: an online plataforma competitiva multiagente para educação em IA”, em Proceedings of 23ª Conferência Anual da ACM sobre Inovação e Tecnologia em Educação em Ciência da Computação. ACM, 2018, pp. 33-38.
[5] ICGA, “Associação Internacional de Jogos de Computador”, https://icga.org/, acesso em 4 de janeiro de 2020.
[6] AlloyTeam, “Codetank,” http://codetank.alloyteam.com/, acessado em 10 de dezembro de 2019.
[7] J. van Eeden, “Riddles.io,” https://riddles.io/, acessado em 18 de junho de 2019.
[8] H. Zhang, G. Gao, W. Li, C. Zhong, W. Yu e C. Wang, “Botzone: A sistema de jogo para educação em inteligência artificial”, em Proceedings da Conferência Internacional sobre Fronteiras na Educação: Informática Ciência e Engenharia da Computação (FECS). O Comitê de Direção do Congresso Mundial de Ciência da Computação, Computação ..., 2012, p. 1.
[9] Wikipedia, “Gomoku-wikipedia,” https://en.wikipedia.org/wiki/Gomoku, acesso em 20 de dezembro de 2019.