基于云技术的远程象棋引擎平台

一、摘要

象棋引擎是象棋爱好者和专业棋手用来分析局面、提高棋力的重要工具。现有的象棋引擎授权方式基本采用绑定硬件，一机一码的授权方式，不仅用户需要自行大笔投资购买高性能的计算机，也不方便外出时使用。本文提出一种基于云技术的远程象棋引擎平台，即“软件即服务(SaaS)”。所有分析计算都在云平台上运行，用户只需一台能联网的设备（电脑，手机皆可）就可以随时随地的调用引擎分析局面。由本地到云的服务模式转换，对于用户和引擎开发者是双赢的。

二、研究背景和意义

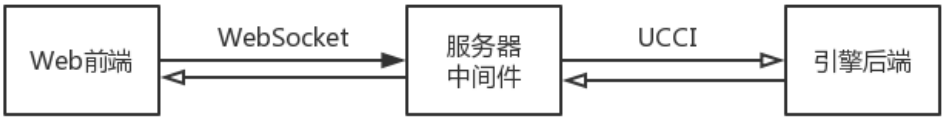
1991 年，象棋引擎的先驱者、人工智能专家虞希舜先生推出了象棋游戏“将族”。其界面美观，且引擎具有一定水平，游戏性很强，推出后不久取得相当的成功。经过众多象棋引擎开发者二十余年的努力，今日的象棋引擎已经具有达到或超过人类顶尖棋手的实力。截至 2018 年，市面上已有多款颇具竞争力的象棋软件供使用者选购，如象棋名手，象棋旋风，阿尔法猫等等。专业象棋引擎的购买者包括不仅包括专业棋手，也包括诸多象棋爱好者。象棋引擎既可以用于直接对弈，也可以对特定局面进行深入的分析，对于提高象棋水平大有裨益。

现有象棋引擎普遍采用单机版的部署方式，并根据购买的许可证数量收费，具体费用与引擎支持的最多核心数有关。以多次获得计算机引擎博弈锦标赛冠军的“象棋名手”为例，该产品采用绑定机器码的授权方式，用户购买一份许可证只可以在一台电脑上运行。如专业选手外出参加比赛，需要在随身携带的笔记本上运行该软件，则需另外购买。此外，由于引擎性能与 CPU 性能紧密相关，家用 CPU 往往无法发挥引擎的最佳性能，所以用户需要额外购买安装服务器处理器（如 Intel Xeon）的专业工作站用于运行象棋引擎，这对用户来说是一笔不小的投入。此外，随着人工智能和深度学习技术的迅速发展以及 AlphaGo 的面世，象棋引擎中融入深度学习技术已是大势所趋。而深度学习相关的运算需要大量的 GPU 并行计算完成，这对大多用户来说显然是难以负担的。而“云平台“技术，则是目前能平衡这一矛盾最行之有效的方案。

本文提出一种基于云技术的象棋引擎平台方案，也是“软件即服务(SaaS)”的一种体现。用户无需自行购买硬件用于运行象棋引擎；用户只需购买一个账户，随后登录服务商提供的分析页面即可使用引擎。服务商（通常也是引擎的开发者自身）根据市场规模租赁或者购置所需的服务器，并借助适当的云技术向用户提供服务。与传统部署方式相比，用户可以大笔节省用于购买硬件的费用，且可以随时随地使用象棋引擎（比如通过手机查询）。对于服务商来说，此种方式可以提供传统方式无法达到的服务质量与体验，提高自身竞争力。并且租用或购买硬件的成本将随用户数量攀升显著下降。此外，提供云服务无需将引擎自身交给用户，可以避免软件自身被逆向分析甚至破解，由此亦可以节省一定的软件保护开支。也就是说，由本地到云的服务模式转换，对于用户和引擎开发者是双赢的。

三、概述

本文提出的云平台主要由三部分组成：即 Web 前端，服务器中间件，以及引擎后端（见下图）。其中 Web 前端接受用户输入（如设置局面，走动棋子），并显示引擎的分析结果（如推演着法，局面分数）。引擎后端就是我们常说的象棋引擎，对局面进行分析与演算，并输出计算结果和局面分值。服务器中间件是沟通两者的桥梁。一方面，服务器中间件接受来自 Web 端的输入，比如设置当前局面，走动棋子等等。另一方面，中间件必须将来自用户的输入（局面表示）翻译为引擎后端能够理解的格式（FEN 串），并通过适当的 UCCI 命令通知引擎进行分析。当引擎返回分析结果和分值时，服务器中间件必须将引擎返回的数据翻译成人类可以理解的格式，比如将编码的棋步（h2e2）转换成通常的记录格式（炮二平五）。



以下分别阐述三部分内容：

1. Web 前端

Web 前端是本平台直接与用户交互的部分。其界面主要包含一个棋盘，用于显示引擎分析信息的区域，以及若干用于设置局面以及控制引擎的按钮。页面加载后，前端首先要求用户输入身份信息，并与服务器中间件进行验证。验证通过后，Web 前端会提示用户已经可以使用引擎并允许用户设置需要分析的局面。分析过程中，若中间件返回分析信息，前端会即时将其显示在屏幕上，供用户查看。

Web 前端是本平台与用户互动的主要界面，所以必须直观而且有效。如图 1、2 所示是本系统的图形界面（分别在桌面浏览器和手机浏览器查看）。其主要部分为棋盘，右侧和下方有若干必要的功能性按钮，在下方显示信息或者引擎分析结果。

本文采用 WebSocket 作为 Web 前端与服务器中间件之间的通讯协议。WebSocket 可以在浏览器与服务器之间建立一个全工的双向连接，即浏览器可以随时向服务器发送请求，服务器也可以随时向浏览器发送信息。而传统浏览器-服务器通信模式中，服务器只能被动的等待浏览器发来请求，进而回复信息，无法主动推送信息。浏览器为了及时或者服务器的输出，只能频繁（例如 1 秒 1 次）的向服务器发送请求，这样会浪费大量资源。相比之下，采用 WebSocket 通信，每当服务器中间件读取到后端引擎有新的分析结果输出，就可以立即向 Web 前端发送。这样不仅效率高，就工程实现上也更简洁。

此外，为适应移动设备的兴起，Web 前端必须同时支持桌面浏览器与手机浏览器，并在两者上都取得良好的显示效果。本文采用响应式网页设计的原则，借助 HTML5 和 CSS 3 的新特性达到这一目标。特别的，为了适应移动端有限的显示空间，前端界面相应的采用竖版设计，即上方摆放棋盘，下方摆放必要的控制按钮并显示分析信息。此外，犹豫移动设备的多样性，前端界面必须自动适应不同大小的屏幕，并在所有大小的屏幕上都显示合适的界面。本系统所有前端界面都通过 CSS3 自动适应屏幕的长宽，并缩放相关元素的大小。

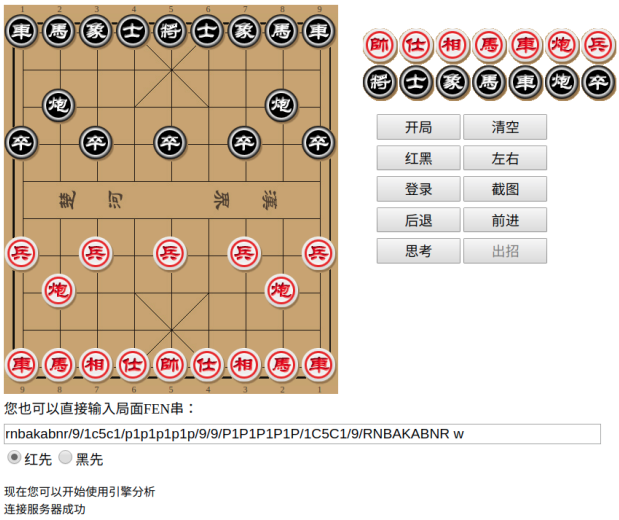


图 1:　云平台电脑端界面

图 2:　云平台移动端界面

2. 服务器中间件

服务器中间件是本平台的业务核心。一方面，中间件必须接受浏览器发起的 WebSocket 请求，读取用户设置的局面和任何引擎控制命令。另一方面，中间件必须与后端引擎建立适当的通讯，将用户设置的局面翻译成引擎可以理解的标准格式（即 FEN 串），并根据用户的意图向引擎后端发送指令（如 ponder 表示思考）。待引擎返回分析结果后，中间件必须对引擎的输出进行解析，将引擎内部的棋谱表示转换成易于人类理解的棋谱。由于引擎内部采用坐标表示棋子，一步棋只包含走子起点和终点的坐标，而人们通常采用的记录方式有时会包括相对位置信息（如前车平三），所以需要额外细心的进行翻译。由于这部分内容与引擎后端紧密结合，故稍后在相应章节内进行介绍。

服务器中间件还必须验证用户身份，只有付费用户才可以使用引擎。并且购买不同产品的用户可以使用不同数量的 CPU 核心，中间件也需要根据用户要求使用的核心数和其被允许使用的核心数进行比较。如果请求的核心数超标，则拒绝其使用或提示升级。

此外，为提供稳定可靠的服务，中间件必须能够根据服务器负载随时扩容，这是云平台独有的优势。事实上，如果服务商为每位用户提供一台专用服务器，这在成本上是无法接受的。云服务的优势在于可以根据负载的多少自动调配资源。比如当前是深夜，平台整体负载较低，则平台自动释放一部分不需要的 CPU 核心，这样成本就会降低。如果当前用户激增，云平台可以向服务器提供商请求更多的 CPU，以满足用户需求。现在主流的云服务厂商，例如亚马逊（aws），谷歌（Google Cloud）均支持多种灵活的控制方式，可以根据当前负载和预期业务需求预期调整所需的 CPU 实例，从而在满足服务需求的前提下最小化运营成本。

3. 引擎后端

引擎后端是实际对局面进行分析的部分。本文并不旨在编写一个新的引擎；现有的引擎，无论是象棋名手这样的商业产品，还是象棋巫师这样的开源项目，都是可用的引擎后端。事实上，引擎后端采用通用的 UCCI 协议接受来外部的输入，只要中间件理解这一协议，既可驱动任何引擎。UCCI 协议提供标准化的通信格式，通信双方（中间件和引擎）必须同时理解该协议才能进行通信。例如 UCCI 协议使用 FEN 串表示局面，中间件必需由棋盘当前局面生成 FEN 串，通过 UCCI 协议传递给引擎后端；引擎后端必需正确接受并转码 FEN 串，将其还原成棋盘表示，方可进行分析计算。

3.1 UCCI 协议与 FEN 串

象棋引擎通常采用 UCCI 协议与外部通信，理解和实现这一协议是本研究的基础。该协议的主要部分是对棋盘的表示，也就是将一个具体的盘面转换成一个程序可以处理的字符串。UCCI 协议采用 FEN 表示局面。

FEN 串至少含有两部分信息，即当前局面和下一步走子的方面，中间用空格隔开。比如图 1 中的棋盘初始局面的表示为“rnbakabnr/9/1c5c1/p1p1p1p1p/9/9/P1P1P1P1P/1C5C1/9/RNBAKABNR w”。其中较长的字符串表示局面的子力位置，w 表示有红方先行。

在子力位置表示中，FEN 对每个棋子赋予一个字母，红黑双方不同，一般红方为大写，黑方为小写。例如“r”表示黑车，“R”表示红车。FEN 的语法从上至下依次表示棋盘上每一行的子力，不同行之间用“/”隔开。在上面的例子中，rnbakabnr 即表示第一行的九枚棋子。第二行没有任何棋子，也就是有 9 个空格，用“9”表示。第三行为“1c5

c1”，也就是首先有一个空格，然后有一枚黑炮，接着五个空格，然后是另一枚黑炮，最后是一个空格。以此类推，即可知道当前局面每一行包含的棋子。

具体的子力对应方式如下表所示：

	红方	黑方
车	R	r
马	N	n
炮	C	C
士	A	a
相（象）	B	B

兵（卒）	P	P
帅（将）	K	k

根据上表，即可实现由棋盘局面到 FEN 串的转换，并将用户输入的局面传送给象棋引擎进行分析。

接下来需要介绍 UCCI 协议中包含的若干指令。这些指令可以用来初始化引擎，设置当前局面，并通知引擎开始分析。下面介绍几条主要的指令。

“ucci”命令可以初始化引擎，接到此命令后引擎会进行一些必要的初始化，并返回一些基本信息，比如当前的引擎设置（CPU 核心数，缓存大小），作者信息等等。引擎正确初始化后会输出‘uciok’，此时引擎即可继续接受新的指令。

“isready”指令用于确认引擎状态，引擎在初始化结束后会返回“readyok”。此时引擎即可进入思考状态。

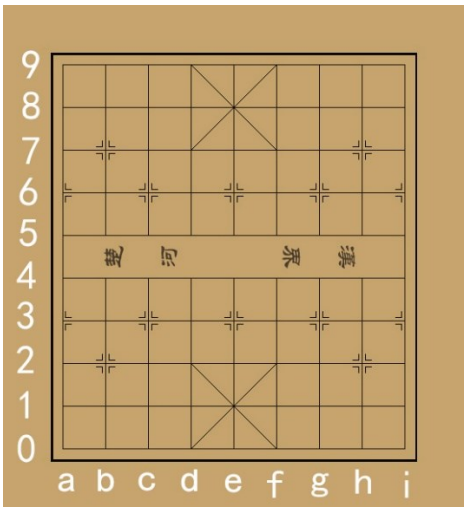
“position fen <fen>”命令将当前局面设置为<fen>表示的局面。例如命令“position fen rnbakabnr/9/1c5c1/p1p1p1p1p/9/9/P1P1P1P1P/1C5C1/9/RNBAKABNR w”就可以告知引擎当前需要分析的局面是图 1 中的初始局面。在本系统中，web 前端会根据用户摆设的局面调用 JavaScript 生成相应的 Fen 串。服务器中间件在接收到 Fen 串后会将该串传递给引擎后端。

通常在设置局面后会接着发送“go ponder”指令让引擎进行后台思考。这时候引擎就会开始分析局面，并返回其正在推演的局面。

3.2 引擎分析着法的解析

引擎在分析过程中，会不断返回当前正在分析推演的变化。具体来说，其中包括当前思考的着法，思考时间，局面分数，以及推演的变化。其中当前思考时间，局面分数等信息都比较简明，无需特殊处理，直接返回给 Web 前端展示给用户即可。但当前着法和推演信息中的走子一般用坐标表示（如 h2 e2），需要翻译成自然语言的记谱方式（如炮二平五）。本节主要介绍这一过程。

象棋内部引擎采用横纵坐标的方式来标记棋盘上的格点。具体来说，竖线从左到右依次记为 a-i （共 9 条），横线依次记为 0-9（共 10 条）。这样，棋盘上 90 个格点中的任何一个都可以由其横纵坐标表示。例如原位的右炮位于第八竖线，第三横线，所以其坐标为 h2.



有了表示棋子位置的方法，表示棋子走动也不复杂—只需记录起点和终点即可。例如着

法“h2e2”，就是 h2 位的炮移动到 e2 位置，翻译成通常所用的记录方法就是“炮二平五”。如果黑方接下来走“马 8 进 7”，则是“h9g7”。如此翻译每一步着法，就可以将引擎返回的走法信息翻译成用户可以理解的记录方法。当引擎输出一串着法的时候，需要进行循环处理，即先解析当前局面，然后更新棋子位置，然后解析下一步，直到最后一步。注意不能在最初的局面上解析所有的着法（而是必须及时更新），因为如果这样，走动过一步的棋子走动第二步的时候就会解析出错。例如，当引擎输出“b0c2 c6c5 g3g4 b9c7 h0g2 a9a8 i0i1 h9g7 a0a1 i9i8 g0e2 g9e7 a1d1 i8d8 b2a2 g6g5 g4g5 e7g5 c3c4 d8d1 i1d1 c5c4 e2c4 g7f5”，其对应的着法为“马八进七 卒 3 进 1 兵三进一 马 2 进 3 马二进三 车 1 进 1 车一进一 马 8 进 7 车九进一 车 9 进 1 相三进五 象 7 进 5 车九平六 车 9 平 4 炮八平九 卒 7 进 1 兵三进一 象 5 进 7 兵七进一 车 4 进 7 车一平六 卒 3 进 1 相五进七 马 7 进 6”。

需要注意的是，解析引擎走法还需要注意一个象棋记录方法的特殊之处。当有两个棋子位于同一竖线时，需要在前面加上“前”“后”予以区分。例如，“前炮平五”“后车退二”等等。但引擎给出的走法信息只包含起点和终点，并不涉及子力的前后位置关系问题。所以在解析走法时必须额外考虑同一竖线上是否有相同的子力，如果有，则需要相应的添加“前”“后”，才能保证输出正确的着法。

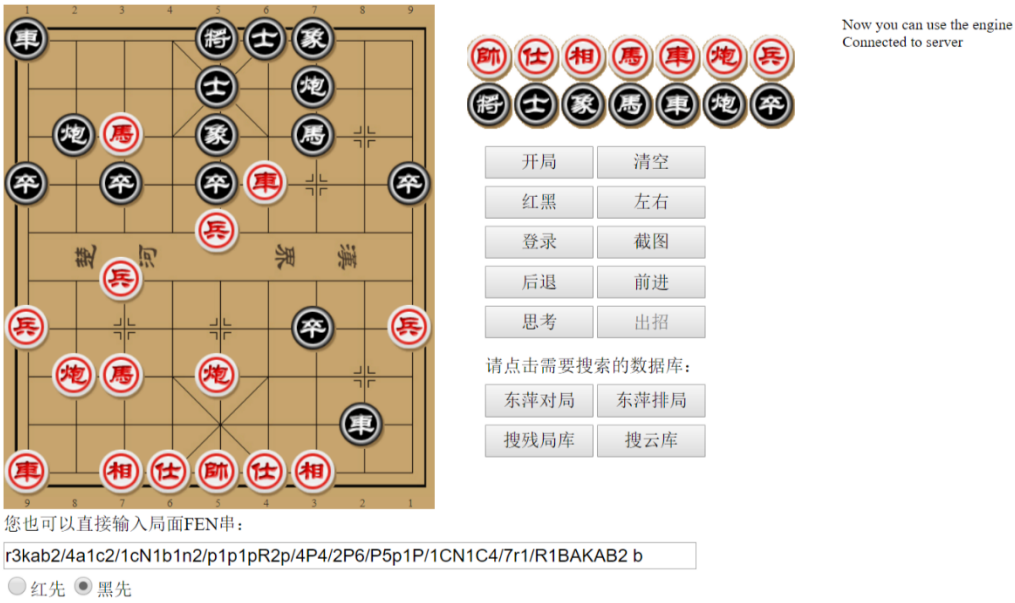
四、系统实现与效果演示

本系统服务器中间件由 Python 实现，Web 前端由 Html+JavaScript+CSS 实现，Web 服务器采用 Nginx，并且同时支持 Linux 和 Windows 系统。以下是系统使用的简单演示：

1. 登录及初始化
   a) 用户购买引擎使用权后，会得到一个密钥（token）。初次访问网站的时候会被要求输入密钥。输入正确密钥后，引擎将会初始化。引擎初始化完毕将会提示用户设置需要分析的局面。

2. 摆设局面
   a) 然后用户将会摆设自己想要分析的局面，具体摆设方式与传统象棋软件类似。除支

持必须的摆子外，本系统还提供左右互换和上下互换等功能，方便用户分析。



3. 分析局面

a) 局面摆设好后即可开始分析，用户将会在页面上看到引擎返回的分析信息。当用户走棋或改变当前局面时，引擎会自动分析新的局面。

五、结论

本文提出了一种基于云技术的远程象棋引擎平台，并介绍了其主要部件的实现方式。云平台相比于传统象棋引擎的分发方式有诸多优势，具有广阔的应用前景。