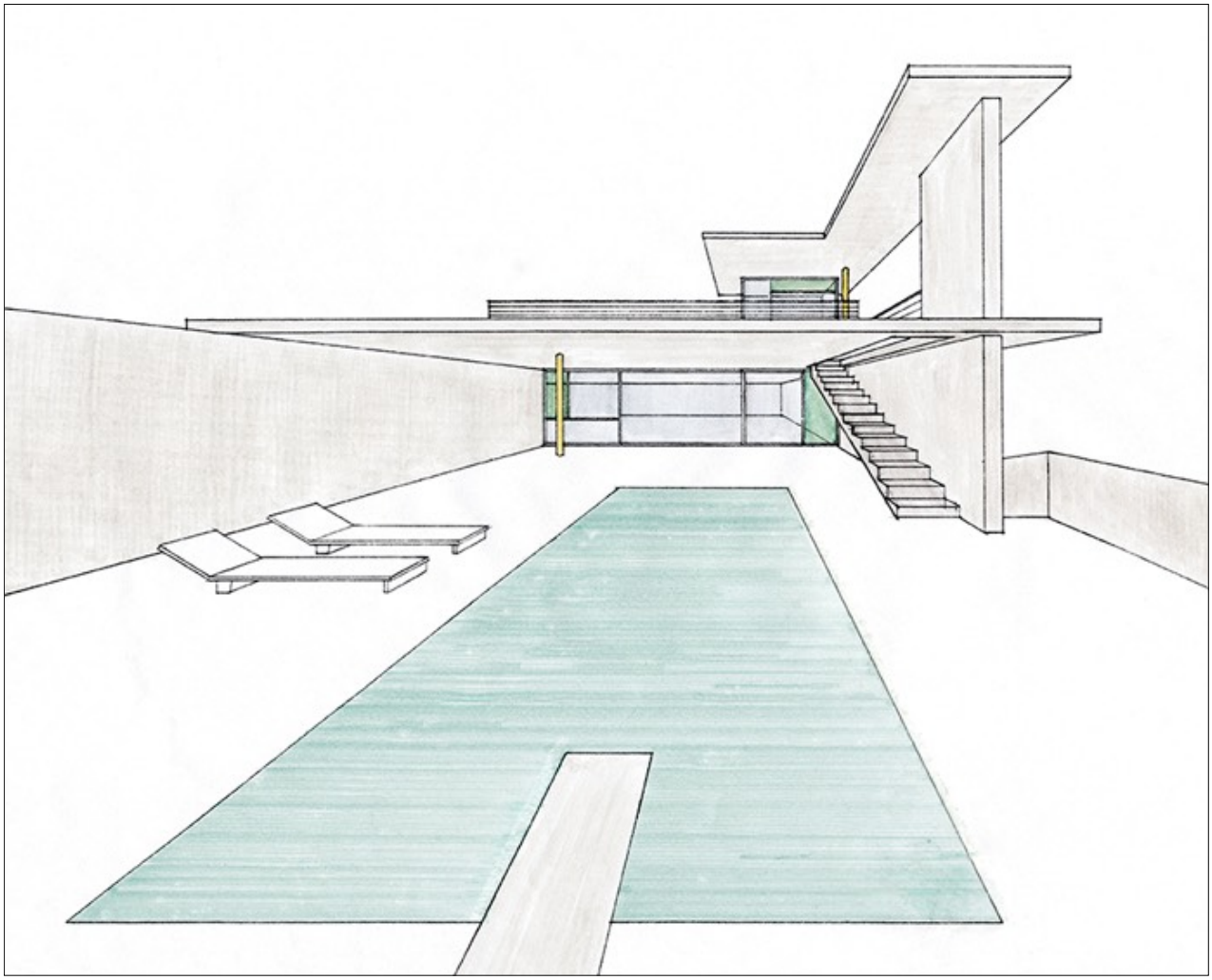

メカ女子将棋　アピール文書

Prepared for: CSA

Prepared by: メカきむりん

2017年3月31日

Proposal number: xxx-xxxx

EXECUTIVE SUMMARY

Objective

「全力で甘える主義♡」をモットーにメカきむりんに全力で甘えますw　女性チーム（一部自称を含む）として女子力をフルに活用したコンピュータ将棋ソフトを目指します。

Goals

順位などはあまり重要じゃなくて（ぉぃ）、みんなで楽しめることを第一に開発しています。コンピュータ将棋を通して、将棋、コンピュータと女子の可愛さの解答を探しています。でも順位は上がいいですぅ〜w

Solution

プログラミング言語としてJulia言語を採用します。将棋プログラムとしては古風なつくりだと思います。（詳しくは後で述べます）

Project Outline

プログラミング言語としてJulia言語を採用します。Julia言語は主にScientificなComputationで使われている言語で、日々進化を遂げています。筆者はその日本での最初期の勃興からJulia言語と付き合ってきました。Rubyなどに近い構文を持ちながら、C++に匹敵する速さを備えている言語です。

今年は特に大きな修正は行ってませんが、選手権まで時間がありますので何かを追加するかもしれません。またリモートでの参加（将棋所はMacのParallels上で動くWindows10で、思考プログラムはアメリカのメタルサーバーで動くUbuntu上で）をします。このために必要な工夫もまたプログラムの一部です。今回はRackspaceのUSリージョンにあるメタルサーバー（VPSではないより速いマシン）を借りる予定です。開発には全面的にMacを活用しています。もちろん、Mac上でも動きます（色々工夫は必要ですが。。。）

CHARACTERISTICS

technologies

反復深化	基本的に将棋プログラムの思考はゲーム木の深さが深くなるほど長くなります。このため決められた思考時間の中でもっとも深く読むために反復深化という技術があります。いえ、実はとても簡単なことで、手を読むときに深さ1、深さ２、。。。とだんだん深くしていって、最後の深さNのところまでで時間切れになったら、その時の最善手を指すというものです。ほとんどのプログラムがこの反復深化を採用していると思います。
ゲーム木探索	これもまたほとんどのプログラムが採用している将棋ソフトの思考方法です。まず、現在の局面（初手なら開始局面）から１手さして変化した局面へと、局面をノード、手をエッジとしたグラフを作り（ほとんどツリーです）現在局面から１手目、２手目とどんどんゲーム木を作っていって、最適な手を探す（つまり現在局面から一番いい局面へと遷移する最善と思われる手を探す）探索をします。ほとんどのケースでAlpha-Beta探索という方法を使います。自分が先手なら先手は後手の局面へ、後手の局面は先手の局面へ、と接続されます。ある局面から一手も合法手がなくなったら（通常は）負けです。（あるいは相手がなくなったら自分の勝ちです）。
合法手生成	まず将棋ソフトは現在局面において次に指せる手（可能手と言います）を駒の効きや打ち駒の場合空いてるマスの情報などから生成します。通常１）普通の動かす手、２）取る手、３）成る手、４）打つ手などを種類別にそれぞれ生成します。このときルール違反やその中でも特に王手放置となる手を取り除きます。こうして残った手を合法手と言います。ただ、通常は合法手を一気に全部生成するのでなく、可能手を作りながら、その中で合法かどうかを探索するときに一手取り出すときに判断することが多いようです。初期局面では30の合法手があります。持ち駒が増えると一気に合法手は増えて、最高で600手くらいになることがあります。通常終盤は200手前後の合法手があります。

Alpha-Beta探索	Knuth先生によって論文として初めてまとめられたAlpha-Beta探索の性質はとても面白いものです。Alpha値とBeta値という、評価関数（後述）の次元を持つ二つのパラメータを使って指定の深さまで探索を行うのですが、このとき小さい値（Alpha値）と大きい値（Beta値）で囲まれた評価値の範囲を指定すると、探索の結果その範囲内に答えがあればその答えの値が、また範囲外だと下限（Alpha値）、上限（Beta値）が返ってくるという性質があります。つまりあらかじめ範囲がどの辺か分かっていれば、かなり範囲を絞って探索できるわけです。範囲が狭いほどBetaカットオフという枝刈り（手を安全に読まない）がたくさん起きて探索時間が短くなります。
枝刈り（Beta cutoff）	Alpha-Beta探索でも少し出ましたが、ウィンドウ（Alpha値〜Beta値の範囲）を狭くすると安全に読まなくていい（＝読まなくても結果に影響が出ない）手を省略することができます。具体的にはある局面の子局面を順に探索していって、ある時点でその後続の子局面を全て読むのをやめます。ということは逆に、早く読むのをやめるように仕向ければいっぱい枝を安全に刈ることができるということに気がつくと思います。つまり子局面へと通じる手の順番（Move Orderと言います）を並び替える、あるいは順にふさわしい順番で生成することができれば、より枝を刈れます。この並び替えのことをムーブオーダリングと言います。取る手などは比較的早く評価されます。こうして最初の１手目で全て枝刈りされる理想的なゲーム木のことをOptimal Treeと呼んでいます。Optimal Treeになるのは大変ですが、これがみんなが目指しているところです。
その他の枝刈り	そのほか一定の条件を満たしたときに「あー読むのや〜めたw」と言ってその局面の子孫局面を読むのをやめることがあります。そのような枝刈りもまた探索の効率を上げるために必要です。メカ女子将棋ではほとんどやっていませんが。。。(^^;)

BitBoard	可能手生成のための仕組みとして、BitBoardというテクニックがありメカ女子将棋も飛車角香車以外はこの仕組みを利用しています。この仕組みは盤面の81マスを81bitの情報として保存し、駒があるかどうかを駒の種類別のbit列で表現します。有るか無いかしか表現できないため、自分の駒の有無と相手の駒の有無のほか、可能手の対象領域である自分の駒以外のマスなどのbit表現を使ってAND/OR/NOT/XORや場合によっては掛け算（！！）で可能手となるbit列を計算します。メカ女子将棋では128bit整数を使っています。
評価関数	ゲーム木の探索において一番末節まで探索されると、その葉ノードの値を暫定的に計算して返す仕組みがないとゲーム木の評価値を計算できません。そのため「ほぼ勝ち・負け」の暫定評価として比較的速く計算できる暫定値を計算します。メカ女子将棋ではBonanzaさんの評価関数をJulia言語に移植していて、データ（fv.bin）だけ使用してJulia言語で評価値を求めることができます。Bonanzaの評価値については他の文献に説明を譲ります（僕が説明しても誤解に基づくものになりそうなので。。。） 通常評価関数はこの評価値を計算するもので、局面と手番（あるいは先後）などを入力すると、16bitの符号付き整数の範囲で(-123みたいに）値が戻ってきます。ゲーム木の評価値は評価関数と誤解されやすいですが、一般的にニコニコなどで評価値と言っているものは現局面からのゲーム木の評価値のことを指し、最善応手手順（もっとも良い手順）の末端の評価値となります（かなり端折って解説すると）。
Aspiration Search	Alpha-Betaのところで出たWindowの性質をうまく使うと、少しだけ速く評価値を出すことができます。これは反復深化と一緒に使います。 まず１手読みで評価値を出します。２手読みでは１手目の評価値の周辺にWindowを設定して読みます。当たったらそれで終わり、外れたら範囲をもっと広げていきます。こうして狭いWindowでの高速な探索をいくつか行うことで、-∞ ~ +∞ のWindowの探索を一遍行うよりも高速に探索できるテクニックがあり、これをアスピレーションサーチと言います。

ヒストリー・ヒューリスティックス	Alpha-Beta探索では良い手が出るとその後の手を安全に枝刈りできることがわかっています。（バグがなければ。。。）そこで、この良い手の開始座標と終了座標（▲76歩なら77と76）を二次元配列に見立て、その交点となる場所にbeta cutoffするたびに小さな値を足していきます。すると、良い手がどれか、ある程度把握することができるようになります。これをムーブオーダリング（読む手の順番を入れ替える）時の数的評価の基準として利用しようというのがヒストリー・ヒューリスティックスです。 実際にはSEEという技術が一番ムーブオーダリングには効くのですが、メカ女子将棋はまだ入れてません。。。