人狼知能ミニコンテスト@GAT2016

レギュレーション

2016/02/13 Ver1.02

1. GAT ミニ大会における人狼のルール

1.1. 各役職の人数

本大会では，15 人のプレイヤーで村を構成します．（初日は犠牲者がいないため，人狼 BBS（http://www.wolfg.x0.com/）における G 国の 16 人村と同様．）各役職のプレイヤー数は以下の通りです．

- 村人　8 人
- 占い師　1 人
- 霊能者　1 人
- 狩人　1 人
- 人狼　3 人
- 狂人　1 人

1.2. 役職説明

1.2.1. 村人側の役職

村人側の役職は「村人」，「占い師」，「霊能者」，「狩人」の 4 種類存在します．

村人

何も能力を持たない村人側のプレイヤーです．

占い師

1 日の終わりに 1 人のプレイヤーを占い，そのプレイヤーが人間であるか人狼であるか知ることが出来ま

す.

霊能者

あるプレイヤーが追放された際に，そのプレイヤーが人間であるか人狼であるか知ることが出来ます.

狩人

1 日の終わりに自分以外の 1 人のプレイヤーを護衛し，そのプレイヤーを人狼の襲撃から守ることが出来ます．本大会では，襲撃が無かった場合に手応え（意図的襲撃ミスなのか，護衛が成功したのか）を得ることは出来ません.

1.2.2. 人狼側の役職

人狼側の役職は「人狼」，「狂人」の 2 種類存在します.

人狼

1 日の終わりに各人狼は，人間を 1 人選択して襲撃投票し，最も多く襲撃投票されたプレイヤーを襲撃します．また，人狼だけが聞くことの出来る「囁き」で，村人に隠れて人狼同士会話することが出来ます.

狂人

村人と同じく能力は何も無い人間ですが，人狼の勝利を目指して行動します．占い師や霊能者の能力では人間と判定されます.

2. 予選，本戦の試合方式

2.1. 予選

予選は，参加者の全プレイヤーのゲーム回数が一定回数以上となるまで以下の試行を繰り返し，平均取得ポイントの上位 15 プレイヤーを選出します.

（※ゲームの回数は時間の許す限り多く行うこととします．）

1 試合の流れ

全参加者のプレイヤーから 15 プレイヤーをランダムに選択して村を構成する．各プレイヤーにランダムに役職を振り分けゲームを行う（各役職の人数は 1．1 節参照）．勝利チームの各プレイヤーに 1 ポイント与える.

※100 対戦毎にプレイヤークラスのインスタンスを破棄し，新たにインスタンス生成を行います．(CEDEC2015 とは異なるのでご注意ください)

予選に参加する場合は，全役職のプレイヤーを実装したソースコードを提出してください（提出方法は「提出方法（3 章）」参照）．また，複数名でのチーム参加も可能です．

2.2. 本戦

予選で選出された 15 プレイヤーに対し、各役職同士の総当りもしくはそれに近似した手法で勝率の多いものを決める．

3. 提出方法

3.1. 参加登録（〆切：2016/2/27 23:59:59）

本大会に参加を希望される方は，人狼知能コンテストエントリーページ（http://contest.aiwolf.org/）より，アカウントを作成してください．

アカウントの作成を持って参加登録とします．

3.2. エージェント提出（〆切：2016/3/5 23:59:59）

参加登録後，全役職のプレイヤーを実装した実行 jar ファイルを同ページより提出して下さい．

C#の方は dll ファイルを提出してください．

なお，提出できるファイルは jar ファイル，dll ファイル一つになります．

Python で提出する方は別途ご相談ください．

4. 禁止事項

本大会では，以下の項目を禁止します．以下の項目に反したプログラムと実行中にエラーを出力するプログラムは予選不戦敗，もしくは個別に連絡を差し上げた上で対処致します．

- TemplateTalkFactory, TemplateWhisperFactory で生成不可能な発話の利用
- ファイルへの書き込み（ファイルの読み込みは条件付きで可）
- ネットワークへの接続
- スレッドの立ち上げ
- 別プロセスでのプログラムの実行

・ 各メソッドにおいて 100ms 以上の計算時間を要するプログラム（多少の誤差は大目に見ます）

ファイル読み込みについて

本大会においては，ファイルの読み込みは禁止とします．
ただし，jar ファイルに含まれるリソースの読み込みは可能です．

5. プレイヤーのプログラム実装について

この人狼ゲームは，http://www.aiwolf.org/server/ 内の aiwolf-ver0.3.x でプレイすることが出来ます．ゲームの起動方法は同ページ内の「AIWolf ゲームサーバ起動方法」を参照してください．
AIWolfCommon.jar 内の org.aiwolf.common.data.Player インターフェースを継承したプログラムがプレイヤーとしてゲームに参加することが出来ます．

5.1. Player インターフェースの実装すべきメソッド（JAVA 版）

Player インターフェースを継承したクラスは 11 個のメソッドを実装する必要があります．これらのメソッドは以下の 4 種類に分類されます．

- 情報処理メソッド： initialize, update, dayStart, finish
- 対象指定メソッド： vote, attack, guard, divine
- 会話メソッド： talk, whisper
- 命名メソッド： getName

5.1.1. 情報処理メソッド（initialize, update, dayStart, finish）

これらは情報を処理するためのメソッドであり，何も戻り値を返す必要がありません．

initialize(GameInfo, GameSetting)

ゲーム開始時に一度だけ呼ばれます．引数として現在のゲーム状態を表す GameInfo とゲームの設定（各役職の人数等）を表す GameSetting が与えられます．（GameInfo と GameSetting については○○参照）

update(GameInfo)

initialize 以外の全てのメソッドの前に呼ばれます．引数としてゲーム内の最新の情報を含んだ GameInfo が与えられます．finish()の前に呼ばれる時のみ，全プレイヤーの役職情報を含んだ GameInfo が与えられます．

dayStart()

毎日の始めに一度だけ呼ばれます．

finish()

ゲーム終了時に呼ばれます．

5.1.2. 対象指定メソッド（vote, attack, guard, divine）

対象となる Agent を戻り値として返すメソッドです．attack, guard, divine は特定の役職のプレイヤーの場合のみ呼ばれるメソッドです．

vote()

その日に投票する対象プレイヤーを返します．

attack()

人狼のプレイヤーのみ呼ばれるメソッドです．その日に襲撃投票する対象プレイヤーを返します．

guard()

狩人のプレイヤーのみ呼ばれるメソッドです．その日に護衛する対象プレイヤーを返します．

divine()

占い師のプレイヤーのみ呼ばれるメソッドです．その日に占う対象プレイヤーを返します．

5.1.3. 会話メソッド（talk, whiper）

発話する内容（String 型）を返すメソッドです．whisper は人狼のプレイヤーの場合のみ呼ばれます．

talk()

村全体に対して発話する内容を返します．本大会で用いる発話は org.aiwolf.client.lib.TemplateTalkFactory で生成出来る発話のみです．（詳しくは「発話可能な内容（4.3 章）」を参照）

whisper()

人狼のプレイヤーのみ呼ばれるメソッドです．人狼だけに対して発話する内容を返します．この発話内容は人狼以外のプレイヤーに公開されることはありません．本大会で用いる発話は org.aiwolf.client.lib.TemplateWhisperFactory で生成出来る発話のみです．（詳しくは「発

話可能な内容（4.3 章）」を参照）

5.1.4. 命名メソッド（getName）

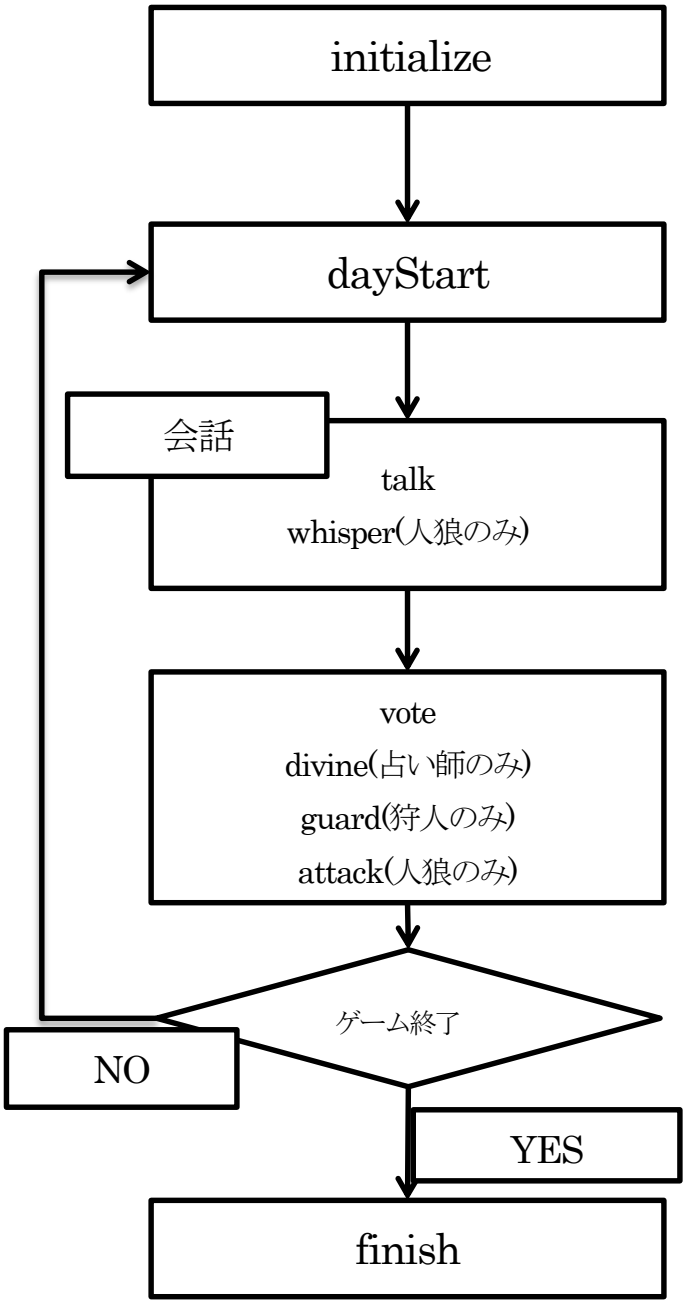
getName()

プレイヤーの名前（String 型）を返します．ゲーム実行時のログに名前が反映されます．

getName では人狼知能コンテストページで登録した「アカウント名」を返すようにしてください．それ以外の文字列を返した場合，決勝進出の対象とならない場合があります．

5.2. 各メソッドの呼ばれるタイミング

4.1.節で挙げた 11 個のメソッドは以下のフローで呼び出されます（getName は除く）． update は省略していますが，initialize 以外の全てのメソッドの前に呼び出されます． また，会話の詳しい流れは 3.2.1 節を参照してください．



5.3. 発話可能な内容

本大会では，TemplateTalkFactory, TemplateWhisperFactory で生成可能な発話のみで会話を行います．生成可能な発話は以下の 11 つです．

- estimate：プレイヤーA の役職は○○だと思う.
- comingout：私の役職は○○だ.
- divined：プレイヤーA を占った結果，○○（人間 or 人狼）だった.
- inquested：プレイヤーA は霊能の結果，○○（人間 or 人狼）だった.
- guarded：プレイヤーA を護衛した.
- vote：プレイヤーA に投票する.
- attack：プレイヤーA に襲撃投票する. （TemplateWhisperFactory のみ）
- agree：発話 T に同意する.
- disagree：発話 T に反対する.
- over：もう話すことは無い. （全プレイヤーが OVER なら会話フェーズ終了）
- skip：様子見（他のプレイヤーが全員 OVER でも会話フェーズが終了しない）

5.4. Player クラスのパッケージについて

Player クラスは独自の物を作成してください. デフォルトである
org.aiwolf.player.RoleAssignPlayer
などを直接書き換えることは避けてください.

Player クラスは，独自パッケージに配置するようにしてください. 通常パッケージはドメイン名またはメールアドレスを逆から利用して作成します.

たとえば，gm@aiwolf.org というメールアドレスをお持ちの方が MyPlayer という Player クラスを作成する場合，パッケージとして org.aiwolf.gm を指定し，

```
package org.aiwolf.gm;


public class MyPlayer extends AbstractRoleAssignPlayer {


}
```

という書き方になります.
また，ソースコード提出時の「class path:」の欄には，

```
org.aiwolf.gm.MyPlayer
```

と指定してください.

6. 他の言語で参加する場合

Java以外の言語はPython，C#のみ想定しています．これらの言語で参加する場合は，それぞれのライブラリを参照にして下さい．独自のライブラリでも TCP-IP 接続が出来，エージェントが正しく動けば参加することが出来ます．

ただし，ゲームは Linux マシンで動きますので，その点を注意して特殊な環境に特化したプログラムはご遠慮下さい．ゲームサーバ上で動かなかったエージェントは自動的に失格となります．

7. 更新履歴

2016/02/13　　ミニ大会用レギュレーション作成