第3回人狼知能国際大会プロトコル部門レギュレーション

2021/04/03 Ver. 1.2

1. 第3回人狼知能国際大会における人狼ゲームのルール

1.1. 村の構成

本大会では，15人および5人のプレイヤーで村を構成します．各村における役職とその人数は以下の通りです．

15人村の役職	村人	占い師	霊媒師	狩人	人狼	裏切り者
人数	8	1	1	1	3	1

5人村の役職	村人	占い師	人狼	裏切り者
人数	2	1	1	1

1.2. 役職

1.2.1.村人側の役職

- 村人：何も能力を持たない村人側のプレイヤーです．
- 霊媒師：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ったか人狼であったかを知ることができます．
- 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を知ることができます．ただし，占い先が追放された場合は占い結果を得ることができません．したがって，霊媒師がいない場合には，追放されたプレイヤーが人間であるか人狼であるかを知ることはできません．
- 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できます．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）を得ることはできません．

1.2.2.人狼側の役職

- 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できます．
- 裏切り者：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．　当然ながら裏切り者は「囁き」を使えませんし，占い師や霊媒師の能力では人間と判定されます．

1.3. 対話の仕様

1.3.1.ターン制の導入

対話はターン制となります．各プレイヤーはターンごとに一回発言することができます（発言しないことも可能）．ただし，発言の順番はターンごとにランダムに決められ，各プレイヤーは自分の順番より前の発言しか知ることができません．

発言は1日に10回まで行うことが出来ます．ただし，SkipとOverは発言に含まれません．すべてのプレイヤーがOverを発言するか，全てのプレイヤーがSkipをするターンが3回連続すると，その日の昼のフェーズは終了します．また，昼のフェーズは最大20ターンです．20ターン経過した時点で昼のフェーズは終了します．

1.3.2.初日対話

初日に対話はできません．

1.3.3.人狼の囁き

人狼の囁きは襲撃投票前（追放者決定後）にのみ行うことが可能です．昼フェーズの対話と同じようにターン制で行われます．初日は襲撃がありませんが，囁きは可能です．偽CO等をする際に初日の囁きを使って相談することが可能です．

人狼が一人になると囁きは行われません．サーバから囁きのリクエスト自体が来なくなる（Javaの場合whisperメソッド自体が呼ばれなくなる）のでご注意ください．

1.4. 投票と再投票

追放のための投票は昼フェーズの最後に行われます．投票によって決定した追放者の情報は，夜フェーズの占い・護衛・襲撃投票に利用することが可能です．

最多得票者が複数となった場合は1回に限り再投票となります．再投票の前に対話はできません．再投票でも同点だった場合は再投票での最多得票者からランダムに追放者が決定されます．再投票では，最多得票者も投票権を持ち，投票者は最多得票者以外にも投票可能です．

襲撃投票でも最多得票者が複数となった場合は1回に限り再投票となります．再投票の前に囁くことはできません．

1.5. 特殊能力

- 占い：占い師は夜のフェーズで占い先を指定して，そのエージェントが人狼かどうかを知ることが出来ます．前述のように占い前に追放者を知ることが出来ます．占いは初日にも行うことが出来ます．
- 霊媒：霊媒師は前日追放されたエージェントが人狼であったかどうかを知ることが出来ます．追放者のいない0日目，1日目は何の情報も得られません．
- 護衛：自分以外の生きているエージェントを護衛先として指定することで人狼の襲撃から守ることが出来ます．護衛先には死者を選ぶことも出来ますが，その場合は何も起こりません．護衛先決定の前にその日の追放者を知ることが出来ます．

2. 予選，本戦の試合方式

2.1. 試合

本大会では固定メンバーによる100回のゲームを1試合とし，これを単位として対戦を行います．1試合の流れは以下の通りです．

1. 全参加チームから5または15チームをランダムに選択して村を構成する．
2. 村の各チームにランダムに役職を振り分け1ゲームを行う．
3. 2.を100回繰り返す．

なお，各チームのエージェントのインスタンスは1試合ごとに破棄され，試合ごとに各チームの番号(AgentIdx)はランダムに割り振られます．言い換えると，1試合の100ゲームの間は

同じインスタンスなのでゲームをまたいだ情報のやり取りが可能で，さらに番号も固定（指定席）です．

2.2. 予備対戦

参加チームは予選および本戦の前に予備対戦を行うことができます．チームがエージェントを登録すると，一日に数回行われる予備対戦のメンバーに追加されます．予備対戦では，各チームが最低1試合(100ゲーム)を行い，エラー無く動いたかどうかが確認されます．自チームが参加した予備対戦のゲームログはダウンロード可能です．提出したエージェントがサーバ上で正しく動作するかどうかの確認やエージェントの改良に予備対戦をご利用ください．

2.3. 予選

予選では，参加全チームの試合回数が一定回数以上となるまで試合を繰り返し，勝率上位15チームを選出します．順位の有意性を高めるため，試合は時間の許す限り多く行います．

2.4. 本戦

予選で選出された15チームに対し、ランダムに役職を割り当てながら試合を多数回行い，勝率によって順位を決定します．

3. エージェント提出方法

3.1. 参加登録

本大会に参加を希望される方（チーム）は，人狼知能大会のウェブサイト(http://contest.aiwolf.org/)にアクセスし，まずはアカウントを作成してください．アカウントの作成をもって参加登録となります．

3.2. エージェントの提出

参加登録後，すべての役職を実装したエージェントのファイルを大会ウェブサイトの「マイページ」より提出（アップロード）して下さい．提出するファイルは実装に用いるプログラミング言語によって以下のように異なります．

3.2.1.Javaエージェント

提出可能なファイルはjarファイル一つです．機械学習などのライブラリなどを利用する場合はjarファイルの中に含めておいてください．自動的にクラスパスに追加されます．

データファイルなどはjarファイルの中に含めておいて，読み込むことができます．たとえば，zipファイル内の/data/hoge.datを読み込みたい場合は，

```
InputStream is = getClass().getClassLoader().getResourceAsStream("data/hoge.txt");
```

と書くことで，ファイルへのInputStreamを得ることができます．

なお，aiwolf-client.jar, aiwolf-server.jar, aiwolf-common.jar, aiwolf-viewer.jar, jsonic-xxxx.jarを含めているとクラス名の衝突（コンフリクト）が発生し，正しく動作しない可能性があるため，お控え下さい．

対戦サーバにおけるJavaのバージョンは11です．それ以降のバージョンを利用してエージェントを作成した場合，正常に動作しない可能性があります．

3.2.2.C#エージェント

提出可能なファイルはdllファイル一つあるいはzipファイル一つです.

エージェントが単一のdllファイルの場合はそのファイルを提出してください. この場合, ファイル名の制限はありません.

エージェントが複数dllファイルで構成されている場合は, それらをzipファイルに固めたものを提出してください. この場合, プレイヤークラスの入ったdllファイルは, 必ず「チーム名.dll」という名前にしてください. 例えば, チーム名がtoriならば, 「tori.dll」という名前でプレイヤークラスの入ったdllファイルを作成しないと起動しません.

データファイルなどは埋め込みリソースとしてアセンブリに埋め込んでおけば, 実行時にAssembly.GetManifestResourceStreamメソッドでStreamを得ることができます.

3.2.3.Pythonエージェント

提出可能なファイルはzipファイルです.

提出時には起動スクリプトファイル名を指定してください. また, 起動スクリプトファイルはzipファイル内にチーム名と同じディレクトリを作成し, その下に配置してください. 例えば, 「team_hogehage」というチーム名で登録し, 起動スクリプトファイル名が「start_mokemoke.py」だとすると,

team_hogehage/start_mokemoke.py

という形で, team_hogehageディレクトリ内にstart_mokemoke.pyを配置することになります. そして, 提出時に指定する起動スクリプトファイル名にはディレクトリ名は含めずstart_mokemoke.pyと指定します.

起動スクリプト同様, すべてのスクリプトファイルはチーム名ディレクトリ以下に配置してください. それ以外の場所に配置した場合は起動しない可能性があります. また, たとえ決勝トーナメントに進出してもレギュレーション違反として失格になることがあります.

getName関数が大会のウェブサイトで登録した「チーム名」を返すようにしてください. 他の文字列を返した場合は勝率がゼロとなります.

3.3. ソースファイルとアルゴリズム概要文書の提出

決勝進出チームには, 大会ウェブサイトから提出した実行ファイルの他に, エージェントのソースファイルおよびアルゴリズムの概要を記述した文書の提出が義務付けられています. 実行ファイルとソースファイルが同一であるPythonエージェントも例外ではなく, 改めてソースファイルの提出が必要です. ソースファイルとアルゴリズム概要文書の提出方法は予選の後に連絡します.

これらを提出しなかったチームは失格となります.

4. 禁止事項

本大会では, 以下の項目に該当するエージェントを提出したチームは, 原則として予選不戦敗となります(場合によっては個別に連絡を差し上げた上で対処します).

- 実行中にエラー
- 人狼知能プロトコルに準拠しない(Javaで言うとContentBuilderで生成不可能な)発話文字列を返す
- ファイルへの書き込み(ファイルの読み込みは, Javaエージェントではjarファイル内のファイル, C#エージェントではアセンブリに埋め込まれたリソース, Pythonエージェントではzipファイル内のファイルに限り可)
- 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
- サーバからのリクエストに対する応答時間が100ms以上

5. プレイヤーのプログラム実装について（Java版）

ここではJavaエージェントを例にとって説明します（C#とPythonの場合も基本的な考え方は同じです）.

第3回人狼知能国際大会では人狼知能プラットフォーム0.6.xの利用を想定しています. AIWolfCommon.jar内のorg.aiwolf.common.data.Playerインターフェースを継承したクラス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

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，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.
- 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Agentを返します.
- attack()：人狼のみ呼ばれるメソッドです. その日に襲撃投票する対象Agentを返します.
- guard()：狩人のみ呼ばれるメソッドです. その日に護衛する対象Agentを返します.
- divine()：占い師のみ呼ばれるメソッドです. その日に占う対象Agentを返します.

5.1.3. 会話メソッド（talk, whisper）

発言する内容（String型）を返すメソッドです. whisperは人狼の場合のみ呼ばれます.

- talk()：村全体に対して発言する内容を返します. 本大会で有効な発言内容はorg.aiwolf.client.lib.ContentBuilderで生成出来るStringのみです（詳しくは「5.3 発言可能な内容」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言する

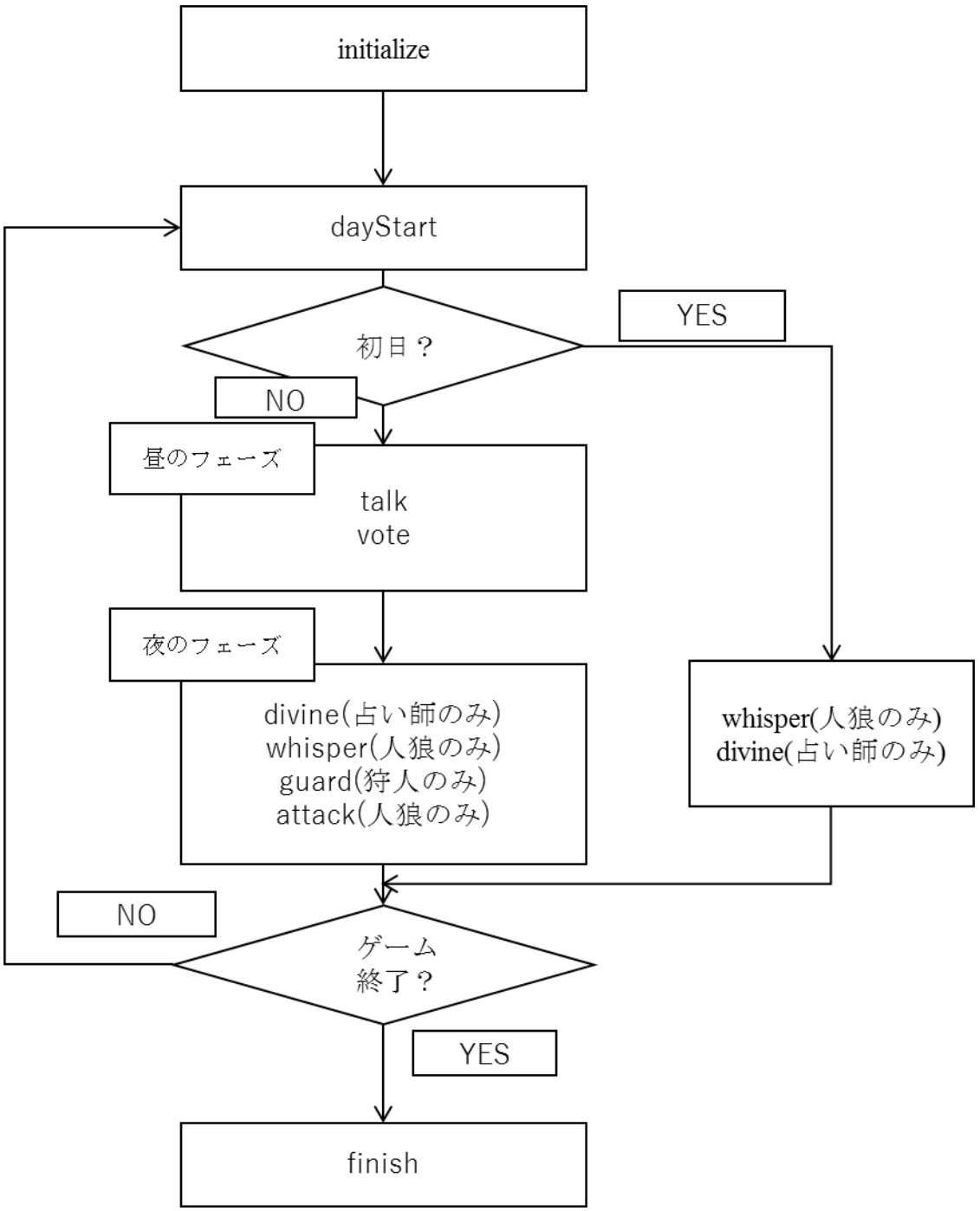
内容を返します．この発言内容は人狼以外のプレイヤーに公開されることはありません．本大会で有効な発言内容はorg.aiwolf.client.lib.ContentBuilderで生成出来るStringのみです．

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．

5.2. 各メソッドの呼ばれるタイミング

getName以外の10個のメソッドは下図のタイミングで呼び出されます．なお，updateはinitialize以外の全てのメソッドの前に呼び出されますので，図中では省略されています．



5.3. 発言可能な内容

本大会では，org.aiwolf.client.lib.ContentBuilderのサブクラスで生成可能な発言のみで会話を行います．生成可能な発言は以下の23種類です．

- estimate：□□が△△の役職は○○だと思う．(EstimateContentBuilder)
- comingout：□□が△△の役職を○○と明かす．(ComingoutContentBuilder)
- divination：□□が△△を占う．(DivinationContentBuilder)
- divined：□□が△△を占った結果，○○（人間or人狼）だった．

(DivinedResultContentBuilder)
- identified：□□の霊能行使の結果，△△は○○（人間or人狼）だった. (IdentContentBuilder)
- guard：□□が△△を護衛する. (GuardCandidateContentBuilder)
- guarded：□□が△△を護衛した. (GuardedAgentContentBuilder)
- vote：□□が△△に投票する. (VoteContentBuilder)
- voted ：□□が△△に投票した. (VotedContentBuilder)
- attack：□□が△△に襲撃投票する.（AttackContentBuilder）
- attacked ：□□が△△を襲撃した. (AttackedContentBuilder)
- agree：□□が発言Tに同意する. (AgreeContentBuilder)
- disagree：□□が発言Tに反対する. (DisagreeContentBuilder)
- request：□□が△△に～を要請する. (RequestContentBuilder)
- inquire：□□が△△に～を照会する. (InquiryContentBuilder)
- because：□□が◇◇の理由で～を主張する. (BecauseContentBuilder)
- and：□□が A, B, … を同時に主張する. (AndContentBuilder)
- or：□□が A, B, … の少なくとも一つを主張する. (OrContentBuilder)
- xor：□□が A, B のどちらかを主張する. (XorContentBuilder)
- not：□□が～を否定する. (NotContentBuilder)
- day：□□がx日目に～であったと主張. (DayContentBuilder)
- 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）(OverContentBuilder)
- skip：様子を見る.（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）(SkipContentBuilder)

5.4. Playerクラスのパッケージについて

Playerクラスはチーム独自の物を作成してください．人狼知能プラットフォーム付属のクラス（org.aiwolf.sample.player.SampleRoleAssignPlayerなど）を直接書き換えることは避けてください.

クラス名の衝突を避けるため，Playerクラスはチーム固有のパッケージに配置するようにしてください．一般に，パッケージ名をドメイン名またはメールアドレスの逆順で命名します．例えば，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;
import org.aiwolf.sample.lib.AbstractRoleAssignPlayer;

public class MyPlayer extends AbstractRoleAssignPlayer {

}
```

という書き方になり，クラスパスはorg.aiwolf.gm.MyPlayerとなります.

6. 他の言語で参加する場合

Java以外の言語はPython，C#(.NET)を想定しています．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ので事前に運営側にご相談(gm@googlegroups.com)ください.

なお，対戦サーバのOSはLinuxなので，その点に注意して，特殊な環境に特化したエージェントでの参加はご遠慮下さい．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．そのようなことを避けるためにも事前の予備予選への参加を強く推奨します．

7. 更新

レギュレーションは予告なく変更される可能性があります．変更された場合はプロジェクトページ(http://aiwolf.org)，開発者メーリングリスト(aiwolfdev@googlegroups.com)あるいはTwitterアカウント(@aiwolf_org)で告知いたしますので，登録あるいはフォローをしておいてください．

更新履歴

2021年4月1日 Ver. 1.0

- 第3回人狼知能国際大会レギュレーション日本語版作成

2021年4月2日 Ver. 1.1

- タイトルに「プロトコル部門」を追加
- アルゴリズム概要文書提出について3.3.に追加

2021年4月3日 Ver. 1.2

- Pythonエージェントがgetname関数でチーム名を返すように注意喚起