DD1363 DD Sec. 4.
Group 20

Per Frost
Marcus Lång
Markus Thurlin
Christer Hedberg
Christopher Engelbrektsson

February 11, 2008

1 Class Resposibility Collaborator Cards

1.1 Input stream

The input stream class inherits from the input stream class provided by the C++ standard library. The class is responsible for encapsulating an input stream, in particular its creation input stream, that of reading data from it, the maintaining of its state and that of returning errors when errors occur. It is constructed from the input stream of the standard library, normal form games can be read from it, extensive games can be read from it, solution types can be read from it, and the input type and the output type are members of it.

1.2 Output stream

The output stream class inherits from of the ouput stream class in the C++ standard library and maintains except for the output stream from which it inherits from an output type. The class is responsible for encapsulating an output stream, in particular creating an It is constructed from an output stream and an output type and solutions can be written to it.

1.3 Input type

The input type is an enumeration of supported input types and has no responsibilities but those of storing a supported input type.

1.4 Output type

The output type is an enumeration of supported output types and has no responsibilities but those of storing a supported output type.

1.5 Solution type

The solution type is an enumeration of supported solution types and has no responsibilities but those of storing a supported output type.

1.6 Normal form game

The normal form game stores a representation of a normal form game, it maintains a player set, a set of strategies a payoff function from the game the cartesian product of the set of strategies to the reals. It provides references to these sets. It is constructed directly from the sets to which it provides references.

1.7 Extensive form game

The extensive game stores a representation of an extensive game, it supplies references to a set of players, to a set of game states, to a set of moves, to a bijective function from the game states to the moves, to a map from the game states to sets of game states, and from the cartesian product of the set of moves to a cartesian product of payoffs corresponding to the players. It provides references to these, and provides a function that takes a solution type and returns a solution.

1.8 Linear program

Linear program maintains a matrix and the transpose of a vector It represents a linear program on canonical form. It provides references to the vector, the matrix and certain functions thereof for the construction of linear program solutions.

1.9 Linear program solution

A linear program solution is a vector and inherits from the vector class provided by the C++ standard library, it is constructed from a linear program.

1.10 Parse error

Parse errors represent exceptions and store a string. A routine enables them to be written to the ouput stream standard error.

1.11 Solution

A solution is a base class for solutions that may be written. It is responsible for storing a solution and for being writeable to an output stream, why it provides methods so that the ouput stream can write it on any of the enumerated output formats.

1.12 Subgame perfect equilibrium

Subgame perfect equilibrium inherits from solution and stores a subgame perfect equilibrium. It provides routines for being written to output stream.

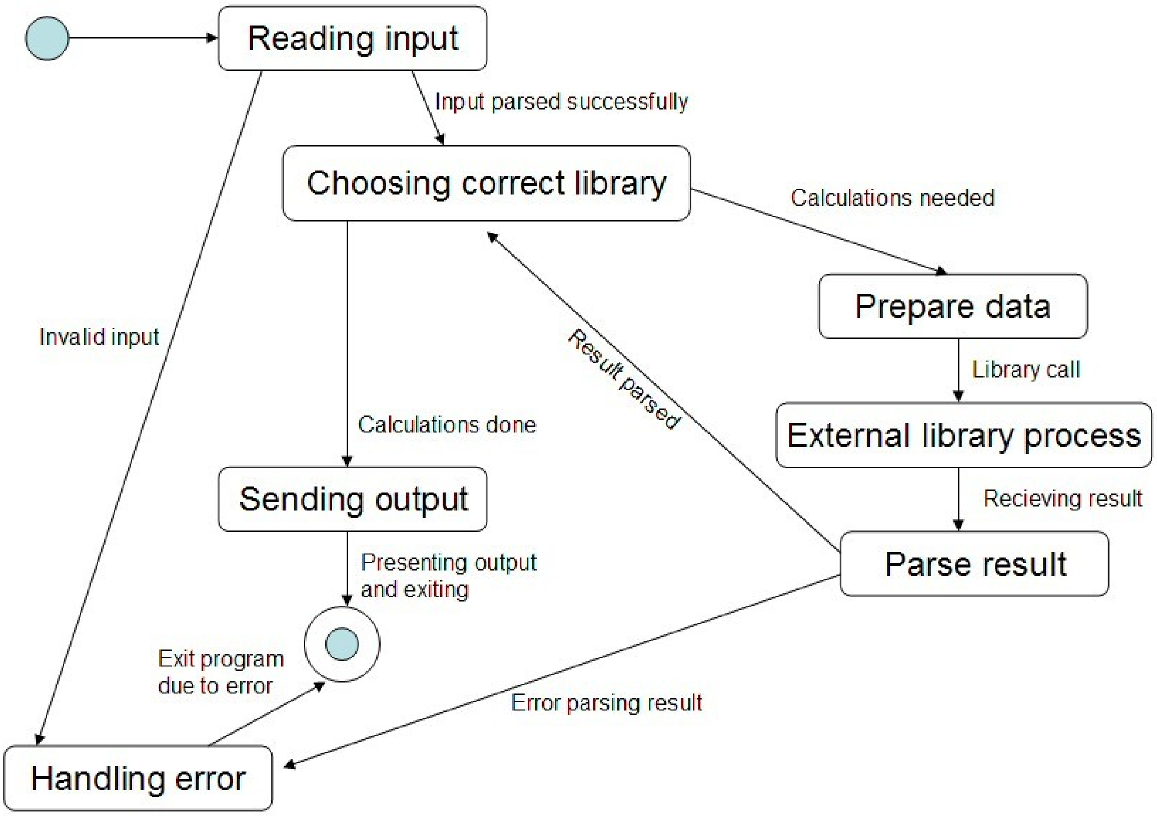
1.13 Nash equilibrium

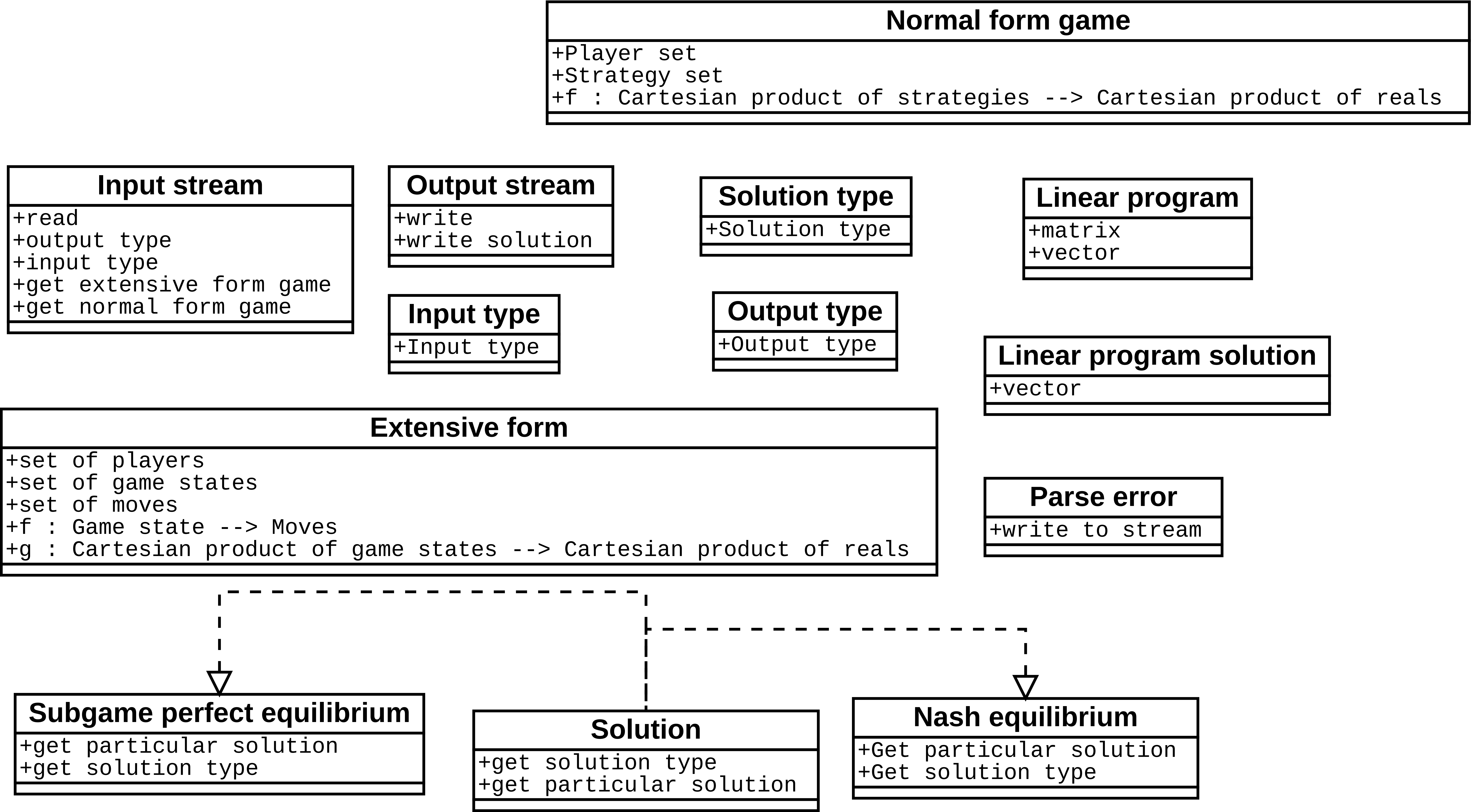
Nash equilibrium inherits from from solution and stores a nash equilibrium, it provides routines for being written to output stream.

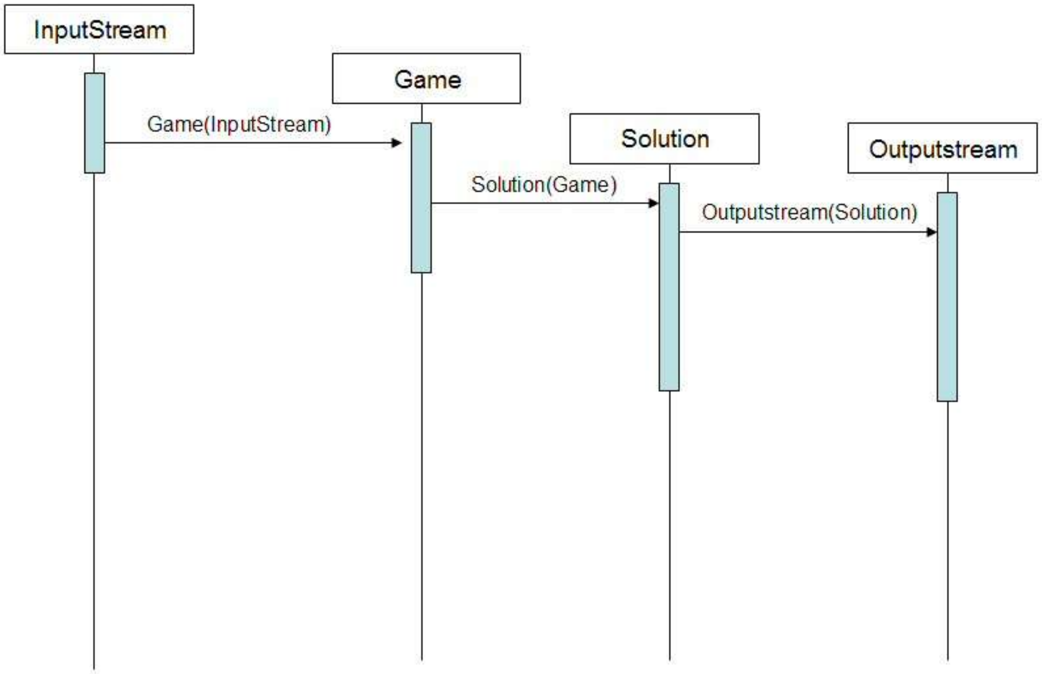
2 Class diagrams

5.3 State Charts

Typical usage




Normal form game
+Player set
+Strategy set
+f : Cartesian product of strategies --> Cartesian product of reals
Input stream
+read
+output type
+input type
+get extensive form game
+get normal form game
Output stream
+write
+write solution
Solution type
+Solution type
Linear program
+matrix
+vector
Input type
+Input type
Output type
+Output type
Linear program solution
+vector
Extensive form
+set of players
+set of game states
+set of moves
+f : Game state --> Moves
+g : Cartesian product of game states --> Cartesian product of reals
Parse error
+write to stream
Subgame perfect equilibrium
+get particular solution
+get solution type
Solution
+get solution type
+get particular solution
Nash equilibrium
+Get particular solution
+Get solution type


InputStream
Game
Solution
Outputstream
Game(InputStream)
Solution(Game)
Outputstream(Solution)