



The experiment consisted of three parts: the login-stage, the experiment and an after-survey. The experiment starts with 6 single-player Tower of Hanoi games to enable learning and to induce routine, referred to as “individual decision-making expertise in routine-strategy”. The single-player rounds are followed by 6 three-player Tower of Hanoi games, where the first three multiplayer-games can be solved perfectly by the agents when sticking to the single player strategy.

This chapter is divided into three parts. The first part describes the software used to measure behavior changes in a complex problem-solving game. As two different versions of the software existed, emphasis on software development process will be laid upon and an overview of the evolutionary process of the experiment is described. The second part describes the participants of the study—details about their background, and where they were recruited from are provided. The third part will explain what participants had to do during the experiment, how data was collected and in which order the experiment was structured.

---

## 4.1 Development and Materials

Two different software versions of the experiment exist. The first version was programmed with “zTree”,

*“a software package for developing and carrying out economic experiments.”*

---

**Electronic supplementary material** The online version of this chapter ([https://doi.org/10.1007/978-3-658-33139-9\\_4](https://doi.org/10.1007/978-3-658-33139-9_4)) contains supplementary material, which is available to authorized users.

(Fischbacher, 2007, p. 172). The software running on zTree was developed in cooperation with a local German IT company. The second version was developed from scrap with same company and embedded in an also self-developed framework for behavioral experiments called “Curiosity IO”.

### 4.1.1 Software Development Process

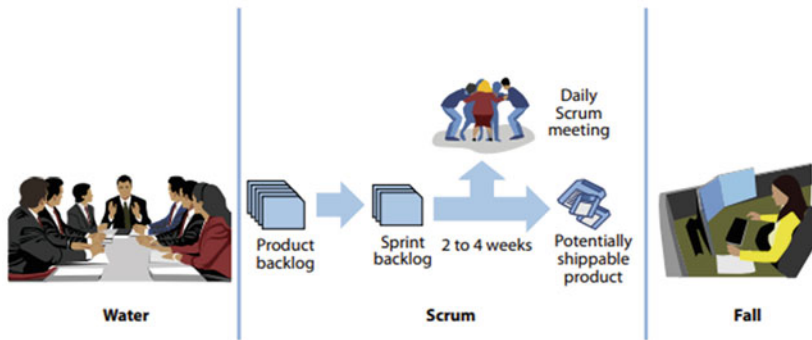
Development processes for both software versions oriented themselves at a combination of the classic “Waterfall” software development process and “scrum”, using “Waterfall” as a framework, and embedding e.g. the face-to-face meetings suggested by scrum.

The software development method scrum

*“assumes that the systems development process is an unpredictable, complicated process that can only be roughly described as an overall progression.”*

(Schwaber, 1997, p. 1). “Water-Scrum-Fall” is a hybrid approach, where “Hybrid Agile methods are a reality in most Agile implementations” (West, 2011, p. 9) and the “Water-Scrum-Fall” approach offers a “simple set of principles, working practices, and roles for teams to execute (...) and guidance on team organization and transparency” (West, 2011, p. 11), while not excluding traditional development milestones. Here the “hybrid method in which traditional and agile approaches are combined seemingly provides the “win-win” situation.” (Theocharis, Kuhrmann, Münch, & Diebold, 2015, p. 13). The simple Figure 4.1 from West et al. (2011) precisely shows the software development process for both experiments. After an extensive meeting (“Water”), the IT company developed the software via scrum, with weekly meetings (“Scrum”), and offered support with bug-fixing, and performance testing (“Fall”) (West, 2011, p. 10). The development of Curiosity IO relied less on weekly meetings however and face-to-face meetings were no longer recorded in written form.

All milestones of the software development process are being listed in chronological order in the appendix table “Software Development Milestones” (see annex 1).

**Figure 6** Water-Scrum-Fall Is The Reality

60109

Source: Forrester Research, Inc.

**Figure 4.1** Software development process Water-Scrum-Fall. *Source* West, 2011, p. 10

### 4.1.2 Legacy Version of Experiment

While an experiment using the zTree program had been conducted and data was collected, both program and data were not used. Instead, Curiosity IO was developed, with an identical experiment, and its resulting data had been used for evaluation. The software development process of the zTree experiment, its application and the reasoning process for abolishing this program and its data will be described, before coming to Curiosity IO and its results. As this “failed” experiment led to new insights and improvement ideas concerning Curiosity IO, it is a fundamental cornerstone of the entire theoretical and practical process. As this legacy experiment was conducted in the historic “Hopfenpost” building in Munich, the experiment is referred to as the “Hopfenpost experiment”. The Hopfenpost experiment will first be described, followed by critical analysis and its problems, ultimately leading to the decision to dismiss the experiment. Software documentation of the zTree software is attached in the appendix (see annex 1).

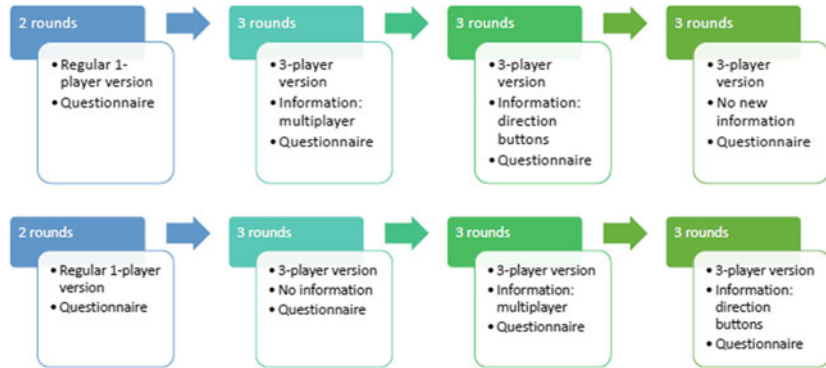
The first experiment was conducted with the zTree software version from 18<sup>th</sup> to 22<sup>th</sup> of June 2018 in a rented room located in the historic “Hopfenpost” Munich. This experiment was conducted “offline” with 264 participants, recruited by two hired companies. The only requirement for all participants was being fluent in German. Before, a website was created, attracting potential college participants with prizes and participation fees. However, this approach deemed to be

ineffective to recruit participants. The two companies recruited 169 female, and 95 male participants for the experiment, with no further restrictions on the participants, such as graduation degrees, age or monthly income. Due to technical difficulties 210 data sets remain useful for analysis. During the five-day timespan, experiments had been conducted from 10 am to 5:30 pm in four groups, in one room, using 18 rented laptops, a 28-port switch, and 3 backup laptops. Due to hot weather, participants had access to water throughout the experiment. Fees were handed out in cash to each participant after the experiment.

Figure 4.2 shows the process of the Hopfenpost experiment, which was conducted via the zTree version of the Tower of Hanoi game. Players were first instructed to take a seat, and to self-report age and sex. Providing an email address was optional, and only necessary, if one was interested in winning a prize. Every experiment was assigned to either group 1 or group 2. Every group's experiment consisted of 4 stages. Before the first stage, participants were orally informed about their task. They had to solve several rounds of "Tower of Hanoi", iteratively answer a questionnaire, and were told about how to correctly use GUI elements in order to do so. They were not deceived by any wrong statement. In the first stage, both groups started with the regular one-player version of ToH, followed by a questionnaire, in order to self-report data on perceived stress (Cohen & Williamson, 1988) and self-reported uncertainty (Clampitt & Williams, 2000). Minor changes were made to both questionnaires to adapt their content to the experiment's context, e.g. replacing "months" by "rounds". The questionnaires were also translated into German and participants were instructed to use the German versions. These two questionnaires were answered after every stage. The second stage started with three rounds of three-player version Tower of Hanoi. Three rounds of three-player version Tower of Hanoi, being referred to as "Tower of Europe", is being played three times in a row as well during stages three and four. The global information between both groups differed in stage two. Global information was always provided orally, and in German. The first group was informed about the fact that they were now sharing control with two other people in this room. The second group did not receive any other information, other than there now being a "Please wait." screen popping up after each input, as output calculation differs. In stage three, the first group was told that the directional buttons did not and will not influence the game at all and in fact, had no influence at all during the past three games, and only change color when being pressed. The second group in stage three was now provided the same information as group 1 during stage two—that they are sharing control with two others in their room. The final stage four offered no new information to group 1 but did provide group

2 with the insight about the “dummy-effect” of the directional buttons, just like group 1 during stage three.

The experiment produced more than 400.csv files of data about the participant’s choices made and self-reported items. However, the experiment was flawed for many reasons, which are now to be explained.



**Figure 4.2** Process model of the legacy “Hopfenpost” experiment. *Source* own source

### 4.1.3 Problems with Legacy Experiment

Throughout the experiment, participants were able to communicate with each other. While the participants were instructed to remain silent, vocal signals such as sighing and moaning, eye contact and clicking sounds could not be avoided, and their potential influence on the data cannot be estimated. Another problem was the high effort in dealing with raw data. As the individual data sets are not linked to the individual seat ID written on each paper, where participants self-reported age and sex, but to the zTree ID, many references would have to be made by hand first. When zTree IDs were “shuffled” in order to randomize groups, it had to be written down by hand, *who-is-who*. In order to being able to truthfully report back to the companies who hired the participants and how many actually arrived, the self-reporting part about sex and age using paper was also done in order to know,

from which company the individual participant was recruited from, as both companies offered different wages. Therefore, the experiment was not fully automated in order to have solid evidence “in paper form”.

Another problem was that zTree is not a very suitable software for GUI heavy experiments. While it is astonishing that a single player and multiplayer Tower of Hanoi game can be programmed using zTree’s rudimentary architecture, it is not graphically convincing. Furthermore, zTree does not offer redundant functions, so that the entire game was programmed with one huge iterative function. This led to the first backup notebook, which ran on an HDD drive, to calculate multiple inputs from more than 9 participants so slowly that participants started complaining, and with more than 10 participants, the entire experiment crashed. A notebook, running on an SSD, had to be used in order to run the experiment with enough efficiency.

During the experiment the actions of each participants could not be monitored. Therefore, the participants had to be monitored by walking behind them, in order to being able to see their computer screens. This was sometimes necessary when participants reported problems with mouse-control, accidentally closed the application or other problems which laid outside of experimental relevance. However, when walking past participants back, some of them reported that being a huge issue and that they will probably behave differently, when being observed from behind. Participants who took a long time to finish the single player version game also reported orally that the presence of others who finished the game faster, made them feel uncomfortable, altering their cognitive stability to the worse.

Due to zTree’s software restrictions, and security concerns, not all inputs could be saved, such as individual presses and clicks on the directional buttons. A group could have solved a stage with only seven actions, but with hundreds of inputs, the latter not being saved, distorting information.

Since global information was reported orally, no recordings of each non-automated part of the experiment exist, and the participants were not isolated, the experiment is hard to defend against accusations of “use of deception”, modeler bias, and noise from communication between participants. In addition, technical problems and bugs resulted in biased data. When a game-group was only containing one or two participants instead of the required three, the software would show unusual behavior, which even crashed the entire experiment. While the latter only happened once, it could have happened every time, when an experiment did not include a number of participants dividable by three.

The questionnaires were slightly changed and translated into German. Therefore, it was not clear whether these questionnaires still validly measured

self-reported stress and uncertainty. The latter survey's quality is debatable, as the study is a rather unknown proceedings manuscript.

The “offline” participant acquisition and entire experimental process was very resource demanding. Due to this reason, and all mentioned problems above, an “online” version of “Tower of Europe”—that is the three-player version of Tower of Hanoi—was developed. While in the beginning the software “oTree” was considered, which is an “online software for implementing interactive experiments in the laboratory” (D. L. Chen, Schonger, & Wickens, 2016, p. 88), which runs on any web-browser without any application requisites, again “oTree” was found to lack graphical requirements, and modern software development language.

#### 4.1.4 Curiosity IO—Structure and Functionality

Software development of Curiosity IO started in November 2018, and was mainly finished August 2019. The software framework contains a number of classic game theoretical experiments such as “Prisoner’s Dilemma”, “Battle of the Sexes”, “Nash Bargaining Game”, “Optional Prisoner Dilemma”, “Public Goods”, “Trust Game”, “Ultimatum Game”, “Dictators Game”, “Public Goods (3 Player)”. It also contains “Flag Run” (Strunz & Chlupsa, 2019), “Dynamic Flag Run”, and “Tower of Hanoi”. Extensive bug-testing and prototype testing of the two main experiments, being “Flag Run” and “Tower of Hanoi”, had taken place. Since the 18<sup>th</sup> January 2019, multiple pre-tests and experiments were conducted. Using the “Flag Run” und “Dynamic Flag Run” game, 13 sessions with 1.459 participants from “Amazon Mechanical Turk” in total were performed, as well as 4 “Tower of Hanoi” sessions with a total of 150 participants—prior to the main experiment. Raw data of all experiments remain saved, and a selection of raw data can instantly be exported in.csv file format. Most screenshots of the software are added to the electronic appendix, and is referred to as “e-appendix” in the following.

Curiosity IO is a framework for online behavior experiments, which can be run on any device and web-browser. Participants can login to an experiment-session by entering the URL <https://www.curiosity-data.com>. Participants then have to type in the experiment’s “Session Code” (see e-appendix chapter3\_1\_participant\_login). When an experiment-session is “open”, and the participant has entered the correct “Session Code”, the participant will automatically begin with the experiment. What comes next, e.g. a label for the participant to provide age and sex, a questionnaire or game theoretical experiment depends on how the experiment was designed by the experimenter.

Experimenters can reach the admin panel via <https://admin.curiosity-data.com>. It is password protected (see e-appendix chapter3\_2\_experimenter\_login). After entering the correct password, the main menu of Curiosity IO is unlocked. It consists of three panels: “Sessions—Create and analyze test runs”, “Level Editor—Create and edit level configurations” and “Survey Editor—Create and edit surveys” (see e-appendix chapter3\_3\_main-menu).

Choosing “Sessions” leads to a list of all performed experiments (see e-appendix chapter3\_4\_gamesessions). The list displays the experiment’s name, type of experiment, date, number of users, and the session code. Pressing the red “X” icon on the top right will lead back to the main menu. Choosing the icon “New Session” will open up the “Create Session” screen (see e-appendix chapter3\_5\_session).

The “Create Session” popup offers various options to design a session. Entire “experiments” are referred to as “sessions”. It can be closed with the red “X” icon, a “Cancel” button, and a session can be created pressing a “Save” button. In the first two empty text fields, a “Session Name” and a “Player Password”, formerly referred to as “Session Code” is to be entered. A checkbox enables the experimenter to activate or disable the panel, which enforces the participant to self-report their sex and age, before being able to start with a session. Each session can consist of three parts: a pre-survey, an experiment and an after-survey. Pre- and after-survey are questionnaires, which can be designed in the “Survey Editor”. Experiments include all listed game-theoretical experiments, as well as “Flag Run”, “Dynamic Flag Run” and “Tower of Hanoi”. All these experiments can be chosen from the dropdown menu “Select Game”. A session can also consist only of surveys—in this case “No Game” has to be chosen.

Depending on the chosen experiment, various options are now enabled, disabled or added. The two dropdown menus “Select Pre Survey” and “Select Post Survey” are always enabled, and list all pre-defined questionnaires created via the “Survey Editor”. When the experiments “Flag Run” or “Dynamic Flag Run” are selected, the dropdown menu “Select Level Configuration” is enabled, and lists all pre-defined level configurations created via the “Level Editor”. When the experiment “Tower of Hanoi” is chosen, multiple elements are added: a button “Add TOH Configuration”, three empty text-fields named “Single-Player Timer”, “Multi-Player Timer”, “Give-Up Timer” and a checkbox labelled “Bot” (see e-appendix chapter3\_11\_createsession\_2). Selecting “Add TOH Configuration” opens makes the “TOH CONFIG” popup appear (see e-appendix chapter3\_12\_tohconfig\_1).

The “TOH CONFIG” popup can be closed with the red “X” icon, a “Cancel” button, and a “Tower of Hanoi” procedure can be created pressing the “Save”



button. Such a procedure has to consist of at least one group. Each group can experience a different procedure. Participants will be put into groups (experiment-group) automatically, without being explicitly informed, by the group number. This will be explained later in greater detail. “Add Group” adds a new group, which can be deleted by pressing the “Delete” icon or edited by clicking on its entry label (see e-appendix chapter3\_13\_tohconfig\_2). When editing a group, consisting of at least one “game”, a list of its games are displayed (see e-appendix chapter3\_14\_tohconfig\_3). The group “edit” menu can be closed with the red “X” icon, a “Cancel” button, and a group procedure can be created pressing the “Save” button. One game is always filled in by default. The edit list contains information about the game ID, number of discs used (min. 3 to max. 10), starting state, goal state, single- or three-player, and whether or not help-text and popup are used. Each “Tower of Hanoi” game consists of three rods. While there exist ToH experiments with more than three rods, Curiosity IO does not offer more than three for the time being. “Add Game” adds a new game, which can be deleted by pressing the “Delete” icon or edited by clicking on its entry label. When editing a game, popup menus and text fields can be used to alter number of discs, start state, goal state, single- or three-player, and whether a popup should be visible (see e-appendix chapter3\_15\_tohconfig\_3). Help-text and popup-text can be entered via the empty text-fields. When a popup is active, it has to be closed by the participant, in order to engage with the experiment. This can be used in order to announce help-text changes or report other important information to the participant.

Going back to the “Create Session” menu, various timers can be set. The “Single-Player Timer” is the amount of time in minutes each participant is provided to solve a Single-Player ToH game. The timer is displayed to the participant during the game, even on the popup. When the timer reaches 0, the next level is automatically started. The “Multi-Player Timer” is the analogue time for each participant to solve the three-player version of “Tower of Hanoi”, also referred to as “Tower of Europe” (ToE). The ToE timer starts after each participant has provided an input. The “Give-Up Timer” is the number of minutes after which a “Give-Up” button appears during ToE. Since each ToE game requires three participants, these three players are called a “game-group”. When the “Give-Up” button is being pressed by any participant of a game-group, the experiment for the entire game-group ends. When the bot checkbox is marked, it will activate the “bot-system”. For the bot-system to work, “Start Wait” or “In Game Wait” have to be filled with an integer. When the ToH experiment comes with ToE games, game-groups of three players are required.

Considering an experiment has only one experiment-group: When participants join such an experiment, they have to wait until two other participants have joined. Those three participants are then put automatically in one game-group, when this experiment has only one experiment-group. This game-group is then automatically assigned to the first experiment-group. The next three participants who joined will be assigned to the second game-group. This game-group will automatically join the next experiment-group, if such an experiment-group exists. “Start Wait” indicates the number of minutes a player has to wait before bots will fill the game-group with either one or two bots. This feature was implemented so that MTurks do not have to wait too long, in order to still offer ethical pay.

Considering an experiment has more than one experiment-group: When participants join such an experiment, they will join, in successive order, each experiment-group.

“In Game Wait” is the number of minutes a player has to wait before bots will fill the game-group with either one or two bots during the game. This feature was implemented so that participants of a game-group could still end the experiment, when a game-group co-player disconnected. This is a good alternative to the more rigid “Give-Up” button concept, as it still allows to produce decision-making data of the entire experiment.

Bots only solve ToE games. Each bot is a simple algorithm using a “random-function”. A “random-function” is usually “pseudo-random”, as it uses various uncorrelated signals of a computer to produce e.g. a random integer. In this case, using this random-function, the algorithm chooses with a 50%-pseudo-random chance the small disk or with a 50%-pseudo-random chance the middle/large sized disk. When ToH or ToE is played with three rods, only the small or any other disk can be moved. Therefore, it is always a binary choice, with only three exceptions: when all disks are placed on one rod, only the small disk can be moved. After having “chosen” either small or middle/large sized disk, the algorithm chooses by 50%-pseudo-random chances whether or not a disk’s moving distance is either 1 or 2 spaces. As there are only three rods, and disks cannot be moved 0 or 3 spaces, this also is a binary choice. How these two binary choices by three agents lead to a single output in ToE will be explained later.

A click on a listed experiment in the “Sessions” menu opens the session overview (see e-appendix chapter3\_5\_session). The session overview lists all participants. Each session overview displays four icons on the top right of the screen. The leftmost icon triggers an experiment as being “active”. While participants are able to log in to an “inactive” session, and start with a pre-survey (when the session is set to “open”, see below), they will be faced with a waiting screen when reaching an actual experiment, such as “Tower of Hanoi”. Only when an

experiment is set to “active” will the actual experiment start. Therefore, the “active” button does not influence pre-surveys, but only experiments. Once active, an experiment cannot be set to “inactive”. After the leftmost icon, the second icon from the left “opens” or “closes” are session with a single click. When an experiment is “closed”, participants can not join a session by entering their session code. The “Proceed” button at the “Join Session” screen will simple not work. When an experiment is “open”, participants are able to join the session by enter their session code, and can begin e.g. with a pre-survey. When a session is both “active” and “open”, participants can come and join the experiment, without the experimenter having to manually open and close the experiment. This feature has been implemented when experiments have to be conducted over the course of days and using e.g. Amazon Mechanical Turk, where participants might come from different time zones. When experiments are performed with people sitting in one room, the “active” feature is useful, so that participants all start with the experiment simultaneously after having filled out the questionnaire. In order to make sure that no further participants can join an experiment, it can be “locked”. The two buttons combined features are explained in Table 4.1 (own source) for better understanding.

**Table 4.1** All permutations of lock and play buttons with effect explanation

active/inactive button	locked/unlocked button	effect
inactive	locked	Default state. No participant can join session. No already joined participant can start with experiment
inactive	unlocked	Participants can join session, and start with pre-survey. No participant can start with experiment
active	unlocked	Participants can join session. Participants can start with experiment
active	locked	No participant can join session. Already joined participants can start with experiment

The third icon opens the “Game Session” options menu or “Session Configuration”. Depending on the experiment, the options menu might differ. The following will explain the option menu as it appears, when a “Tower of Hanoi” experiment is chosen (see e-appendix chapter3\_16\_sessionconfiguration). The “Session Configuration” can be closed via the red “X” or the “Cancel” button. “Restart Session”

will make each participant start over from the very beginning of the entire session, e.g. participants have to re-do their pre-survey. Data created is saved server-side, but cannot be downloaded via the options menu any longer (see below “Download Level Data or “Download Survey”). “Kick all and restart Session” will have the same effect as “Restart Session”, but all participants will have to rejoin the session via the “Join Session” screen. “Clone Session” will make create an identical copy of a session in the “Game Sessions” list. The new session will appear with the original session name followed by “\_clone”. This comes in handy, when a session is complex and takes much time to create. This way multiple clones with e.g. slightly altered timer times can be tested, before a main experiment is conducted. In order to do so, a session can be edited. This can be done via “Edit Session” in the “Session Configuration” menu. A session cannot be edited as soon as it was set “active”. When an experimenter wants to edit an already “active” session, the session can be cloned first, and then edited. Any session can be deleted by pressing “Delete Session”. A popup will then ask “Are you sure you want to delete this session?”. In order to remove the session from the “Game Sessions” list, this action has to be confirmed by pressing “Proceed”. Choosing “Download Level Data” will download a.csv file with the experiment’s raw data, which differs from game to game. When a “Tower of Hanoi” session included ToH and ToE, a single- and multiplayer.csv will separate raw data from one-player and three-player games. “Download Survey” will download a.csv file with the sessions’ questionnaire raw data, which differs according to surveys’ structure and content.

During an experiment, each participant can be monitored “live”. The “Game Session” overview displays the participant ID, the current level or stage the participant is in, the total playing time, and the playing time in the current stage. Clicking on a listed participant during or after the experiment, will display three different icons (see e-appendix chapter3\_6\_session-user-options). Choosing the leftmost icon displays survey answers, which can also be monitored “live” during the experiment (see e-appendix chapter3\_7\_user-options\_1). The center icon opens the experiment monitoring tool, where each input of the participant during the experiment is listed, and can also be observed “live” during the experiment (see e-appendix chapter3\_8\_user-options\_2). The rightmost icon removes a participant from the experiment. The session overview also displays an icon “Add-BotGroup” when the experiment is of type “Tower of Hanoi” (see e-appendix chapter3\_9\_session-bot). A “bot” is a simple algorithm, which randomly chooses inputs, as explained before. Clicking this icon adds a group of three bots, which will solve any ToE game. This feature was added to receive data about how many steps are required to solve “Tower of Europe” when the agents acts randomly.

This concludes the main menu's "Sessions" part. The "Level editor" is only used for "Flag Run" and "Dynamic Flag Run" games. Since this thesis builds upon raw data from the experiment "Tower of Hanoi", this main menu option is skipped.

The next main menu option is the "Survey Editor". Choosing "Survey Editor" leads to a list of all created surveys (see e-appendix chapter3\_17\_surveylist). The list displays the survey's name, date of creation, last date being modified and last date being used. Pressing the red "X" icon on the top right will lead back to the main menu. Choosing the icon "New Survey" will open up the "Survey Editor" screen (see e-appendix chapter3\_18\_surveyeditor\_1).

The "Survey Editor" can be closed via the red "X" or being closed and saved via the "Save" button. By default, a single question already exists. Each question is listed in the "Survey Editor". Each question can be assigned to a group. Groups can be created, edited and deleted by clicking the "Groups" button. The "Groups" feature has yet only be tested with the "Dynamic Flag Run" game, and will not be explained in further detail. The listed questions can be ordered with the grey arrow buttons, and can be deleted with the trash-bin button. Pressing "New Questions" adds a new question. A question can be designed with right-hand side features. A question can be of type "One Choice", "Multiple Choice", "Scale" or "Free Text".

Each question of type "One Choice" and "Multiple Choice" can be formulated via a text-field and can consist of multiple answers. Pressing "New Answer" will make a text field appear, where the answer can be formulated. Order of answers can be rearranged via two direction buttons. Special features to an answer can be added. Those features have yet only be used in the "Dynamic Flag Run" game, and its description is to be skipped here. An answer can also be deleted via the trash-bin symbol.

Questions of type "Scale" only have one answer. Its lower and upper end, with according description, can be modified via text-fields. An example would be "very low, 1, 10, very high" (see e-appendix chapter3\_19\_surveyeditor\_2). Questions of type "Free Text" will automatically provide a text-field for each participant.

This concludes the main menu's "Survey Editor" part and the entire options available to the experimenter via Curiosity IO admin panel. In the following, a "Tower of Hanoi" experiment from the perspective of a participant is shown, in order to discuss their features and functionality.

### 4.1.5 “Tower of Hanoi” Example Session

The session consists of a pre-survey, a “Tower of Hanoi” experiment and an after-survey. The experiment consists of one experiment-group, consisting of one game-group with one human agent and two bot. The experiment will have two games. One classic single-player ToH game and the three-player version (ToE).

In detail, the pre-survey and after-survey are identical questionnaires consisting of one “One Choice” question with two possible answers and are created using the “Survey Editor” (see e-appendix chapter3\_25\_phdexample\_6). The sessions’ experiment (see e-appendix chapter3\_20\_phdexample\_1 and chapter3\_21\_phdexample\_2) is called “PhD Thesis Example Session” with session code “phd”. Sex and age is chosen to be a mandatory choice for each participant. ToH and ToE timers are set to 1 minute. The bot was activated, and its activation or waiting time set to 1 minute before the experiment, and set to 1 minute during the experiment. The described surveys, named “Done\_Before”, are chosen to be pre- and after-surveys. There is only one experiment-group (see e-appendix chapter3\_26\_phdexample\_7). The experiment-group holds two games, with three discs, starting rod being the leftmost rod (Start == 1). The goal rod of the ToH game is set to be the middle rod (Goal == 2), and the goal rod of the ToE game is set to be the right rod (Goal == 3). Both games have a help text and a popup (see e-appendix chapter3\_22\_phdexample\_3). Help-texts and popup context were set to be different for this example (see e-appendix chapter3\_23\_phdexample\_4 and chapter3\_24\_phdexample\_5).

The session is set to be “inactive” and “open”. The participant types in the session code “phd” to join the session (see e-appendix chapter3\_27\_joinsession). After that the participant provides sex and age (see e-appendix chapter3\_28\_joinsession) and is immediately brought to the pre-survey after submitting these details (see e-appendix chapter3\_29\_phdexample\_10). Upon having provided an answer, the participant is now facing the “waiting screen” (see e-appendix chapter3\_30\_phdexample\_11), as the session’s experiment is set to “inactive”. The experimenter sets the experiment to “active”, which also automatically “closes” the session (see e-appendix chapter3\_31\_phdexample\_12). The participant is still only seeing the “waiting screen”, since two more participants have to join in order to form one game-group. The “Tower of Hanoi” experiment groups participants at the very beginning, even before a ToH game, when at least one ToE game is part of the experiment. Thus, it is defined from the beginning, who will face who in the second ToE game. After 1 minute a bot called “10000” joins the game-group. After another minute a second bot called “10001” joins the game-group (see e-appendix

chapter3\_32\_phdexample\_13). At this moment, three agents are part of the game-group, starting the experiment immediately. The participant no longer sees the “waiting screen” but is looking at the instructions popup, with the text “This is the popup.”, a popup “OK” button to close the popup, the timer displayed on the popup, and the “helptext label” next to the popup (see e-appendix chapter3\_33\_phdexample\_14). After closing the popup by pressing “OK”, the participant can see the actual “Tower of Hanoi” game: a caption titled “Tower of Hanoi”, with the timer now displayed below, three rods placed on platforms, a red marked goal rod with “Goal” written below to also address colorblind participants, and the three disks on the left rod in start-state setup (see e-appendix chapter3\_34\_phdexample\_15). The participant has to press the small disk in order to make the “Steps” buttons, “Direction” buttons and “GO” button appear (see e-appendix chapter3\_35\_phdexample\_16). After having chosen number of steps and direction, the participant can confirm the selection made by pressing “GO”, upon which the resulting state will be displayed. When the participant manages to solve the game by positioning the three disks onto the goal rod or when the timer runs out, the “Level Completed” screen is displayed (see e-appendix chapter3\_36\_phdexample\_17). This screen can be skipped by pressing “Next Level”—however the screen is automatically skipped when the participant reached the “Level Completed” screen not by solving the game but because the timer ran out. After the ToH game the participant now looks at the ToE game, with an instructions popup “This is the second popup”, a “OK” button to close the popup and the “helptext label” with “This is the helptext label with different information now.” written on it (see e-appendix chapter3\_37\_phdexample\_18). No timer is displayed, since the timer starts after all three agents of the same game-group have to provide an input first during a ToE game. This is done as agents of the same game-group might have to wait for their co-agents to reach the ToE game; e.g. when two agents are still playing the first ToH game and the third agent has already reached the second game (ToE), and provides an input selection (disk/number of steps/direction, and pressing “GO”), this agent will face the “Waiting Screen”, until all members of the same game-group have provided an input selection. After closing the popup, the participant can choose an input just like in the first game. This time, the goal rod is the very right one (see e-appendix chapter3\_38\_phdexample\_19). After confirming disk, steps and direction selection, the timer starts (see e-appendix chapter3\_39\_phdexample\_20). The output is the product of the one human and of the two bot agents input selections. After the ToE stage, no “Next Level” screen is displayed, as it was the final game. Instead, the after-survey is shown, which in this example, is identical to the pre-survey (see e-appendix chapter3\_40\_phdexample\_21). When completing

the after-survey, the “Thank You” screen with the participant ID is displayed (see e-appendix chapter3\_41\_phdexample\_22). MTurks for example are instructed to provide the experimenter with this ID, so that the experimenter can check whether or not the MTurk has actually finished the session.

This concludes the example experiment. In the next sub-chapter, data output is described and explained.

#### 4.1.6 Example Session Data Output

Using the options icon in the “Game Session” menu, and pressing “Download Level Data” and “Download Survey” several.csv files are downloaded (see e-appendix chapter3\_42\_phdexample\_23): “PhD\_Thesis\_Example\_Session\_single\_player” (ToH.csv), which contains raw data about all ToH games that were part of the “Tower of Hanoi” experiment. “PhD\_Thesis\_Example\_Session\_three\_player” (ToE.csv), which contains raw data about all ToE games that were part of the “Tower of Hanoi” experiment. “PhD\_Thesis\_Example\_Session\_survey” (Survey.csv), which contains raw data about all surveys that were part of the session. For more efficient statistical analysis of the thesis specific hypotheses, two additional.csv files were added after having conducted the main experiment. Tables explaining their variables are added to the appendix, being referred to as “Master.csv” and “Progress.csv” (see annex 5 and annex 6). The content of Progress.csv and Master.csv is not explained in further detail in this chapter, as their content is explained by the description of independent and dependent variables.

Survey.csv lists several raw data (see e-appendix chapter3\_43\_phdexample\_24), all summed up and explained in the according table in the appendix (see annex 2). All types of raw data being saved in Survey.csv, an explanation of its meaning, and how the raw data looks like in the example Survey.csv output file.

ToH.csv lists several raw data all summed up and explained in the according table in the appendix (see annex 3). The table lists all types of raw data being saved in ToH.csv, an explanation of its meaning, and how the raw data looks like in the example ToH.csv output file. Some raw data will be explained in greater detail in the following sub-chapter.

ToE.csv lists several raw data all summed up and explained in the according table in the appendix (see annex 4). The table lists all types of raw data being saved in ToE.csv, an explanation of its meaning, and how the raw data looks like in the example ToE.csv output file. Long raw data names were cut to save table space. Some raw data will be explained in greater detail in the following sub-chapter.



Tables in the appendix (see annex 2–6) show all raw data being exported to.csv format. The next sub-chapter will explain listed data in more detail, such as how data is created—this is supported by data examples derived from the Curiosity IO example session described in the former chapters.

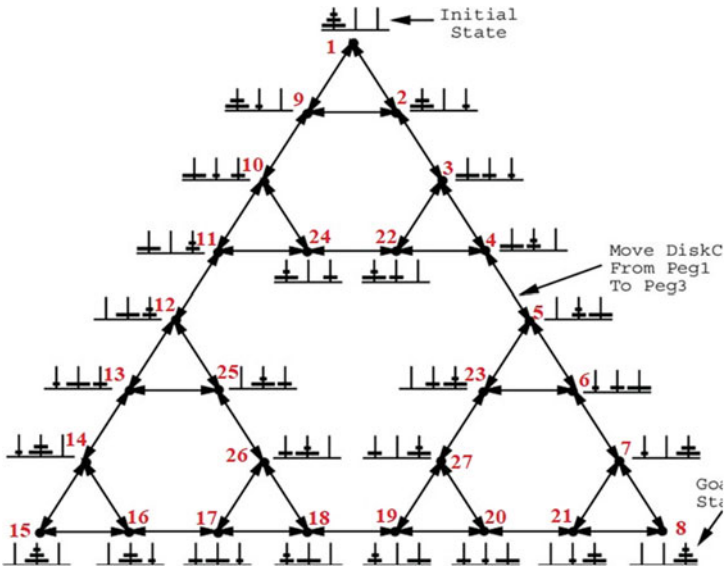
### 4.1.7 Response Time and Input

Response time creation differs for human and bot agents. While response times for bot agents are simulated, as their input order is randomly assigned, response times by humans are always server-dependent. Due to technical issues response times are not reliable enough to make precise statistical analyses. Response times are therefore disregarded.

### 4.1.8 States Derived from State-Space

Data such as “start\_state” in ToH.csv and ToE.csv require a state-space of the experiment. When including operators, the state-space between ToH and ToE differs, as the former only requires a single action to produce an operator, and the latter requires three actions to produce an operator. However, the resulting output states of ToE are identical; if the three actions which produce an operator in ToE are ignored or modelled as being “intrinsic” to the operator, ToH and ToE state spaces are isomorphic.

Figure 4.3 shows the state-space of both ToH and ToE (Knoblock, 2000, p. 3), with integers added to each knot. It consists of 27 knots, representing all possible states, which are visually represented in the “Tower of Hanoi” experiment. It also contains directed, double-headed graphs, representing all possible transitions between states by the according operator. At 24 knots three different operators exist. This is not the case in all possible “start” states (not to be confused with start\_state), which are states 1, 8 and 15. Here, only two operators exist. From each state there exists a path of seven operators leading to states 1, 8 and 15. Since only these three states can be set to goal states, there always exists an ideal path of seven operators towards the goal state.



**Figure 4.3** Tower of Hanoi state space model with their according integers. *Source* Knoblock, 2000, p. 3, red integers added by author

### 4.1.9 Move States

In ToE.csv the “move\_state” data is an integer from 1 to 64 referring to all possible three-player action combinations leading to an operator (see e-appendix chapter3\_Algorithm-States\_new\_appendix\_44). An action consists of a chosen disk and number of steps. Chosen direction is only important when assigning a certain type of logic, which will be explained in a later sub-chapter.

In states 1, 8 and 15 only the small disk can be chosen. In any other state, agents can either choose the small disk or one of the remaining larger disks. With three rods, there is not state where an agent can choose between more than two disks in total. Therefore, each agent is confronted with a binary choice when deciding which disk to move—exceptions are state 1, 8, and 15. Each agents can choose between 1 or 2 steps, at any state, making it a binary choice. Therefore, an individual action can be described by two symbols. “S” representing “Small Disk”, “M” representing “Medium or Large Disk”, 1 representing “One Step” and

2 representing “Two Steps”. Any operator consists of three actions. The operator being referred to by “move\_state” 1 is “S1, S1, S1”, meaning that all three agents have chosen to move the small disk one step. Another example would be “move\_state” 55, with actions “M1, S2, M2”. Here, the first action selection was “Move medium or large disk one space”, the second action selection was “Move small disk two spaces”, and the third action selection was “Move medium or large disk two spaces”.

#### 4.1.10 Operator Output Function

Each set of three actions lead to one specific operator (see e-appendix chapter3\_Algorithm-States\_new\_appendix\_44). The idea of ToE was that each agent held control over the game, and that the direction buttons were not influencing the game at all. The order of information or the order of actions matters for calculating the operator. It was important that no participant would be able to gain advantage over others, meaning that the algorithm was implemented in such a way that no agent would be able to gain more control than other players. This was achieved by restricting communication. As the order of information matters, no agent can be sure at which position its action will be listed. The algorithm was also built in such a way that it can be used for more complex games or state-spaces than “Tower of Hanoi”, e.g. a 4-rod version of “Tower of Hanoi”. The ruleset for the algorithm is derived from a meta-logic and adjusted to the experiment’s degree of complexity. How the algorithm (function) determines the operator (output) by three actions (input) is now explained.

Three inputs are received, e.g. “S1, M2, S2” listed here in chronological order. Only the first two inputs are regarded, and only checked for “choice of disk”, e.g. in this case “S, M”. By doing so “The Decider” and “Direction-Deciders” are obtained. The Decider’s action indicates which disk is being moved and how far it is being moved. Table 4.2 (own source) shows all possible permutations.

By example, the input “S1, M2, S2” leads to “S1” being the “Decider” action, meaning that the small disk will be moved one space. Now, the direction has to be obtained. In order to do so, inputs by the “Direction-Deciders” have to be regarded. Direction-Deciders are the two agents who are both not the “Decider”, e.g. in this example, the 2<sup>nd</sup> and 3<sup>rd</sup> actions are Direction-Decider actions, as the 1<sup>st</sup> action is regarded as Decider action. By doing so, each agents has an influence on the game, with nearly eliminating the chance of any agent using a strategy to always be the Decider or always be the Direction-Decider, also never holding “full power” over disk-, steps- and direction-selection.

**Table 4.2** All possible input combinations leading to resulting „direction-deciders“

Possible “choice of disk” input (first two, by time)	Resulting “Decider” action	Resulting “Direction-Decider” actions
S, S	2 <sup>nd</sup> action committed	1 <sup>st</sup> & 3 <sup>rd</sup> action com
S, M	1 <sup>st</sup> action committed	2 <sup>nd</sup> & 3 <sup>rd</sup> action com
M, S	3 <sup>rd</sup> action committed	1 <sup>st</sup> & 2 <sup>nd</sup> action com
M, M	2 <sup>nd</sup> action committed	1 <sup>st</sup> & 3 <sup>rd</sup> action com

In order to define in which direction a disk moves, the two Direction-Decider inputs are regarded. Now the inputs are only checked for “choice of steps”, in this example “M2, S2” are the Direction-Decider inputs, and the two inputs checked are “2, 2”. The Direction-Deciders’ actions indirectly indicate in which direction the disk is moved, as the Decider’s Choice of Range is also considered. Table 4.3 (own source) shows all possible permutations that decide the direction of the disk.

By mentioned example, Direction-Deciders input “2, 2” and Deciders input “1” leads to the direction of the disk being “left”. The resulting operator equals “move\_state” 29, where the Small Disk is being moved 1 step to the left.

**Table 4.3** All possible input combinations of direction-deciders leading to direction of disk

Possible “choice of steps” input by Direction-Deciders	Possible “choice of steps” input by Decider	Resulting direction
11	1	left
	2	right
12	1	right
	2	left
21	1	right
	2	left
22	1	left
	2	right

The algorithm to determine the operator is purposefully more complex than required to fairly share control, in order to being used for more complex state-spaces as well. Since an operator can lead to an illegal state, such as a bigger disk lying on top of a smaller disk, the next sub-chapter will explain, how an operator leads to the resulting output state, which can be observed by the agents.

For this, a more fundamental definition of a “Tower of Hanoi” game is provided, and differences in handling illegal moves in a ToH and ToE game are shown.

### 4.1.11 State Output Function

When designing the ToE game, several choices had to be made, how closely related ToE should be to ToH. The core idea designing ToE was to “still play a Tower of Hanoi” game. Therefore, the core rules that make a game a “Tower of Hanoi game” had to be defined. Those “axioms” of ToH are:

- A move is a state, which differs from the last state
- No state may show a larger disk lying on a smaller disk
- The game consists of three rods

The first axiom leads to the impossibility to move a disk and bring it back to the original position. The second axiom never allows a move resulting in some state, where a bigger disk lies on top of a smaller disk. The third axiom limits the game's complexity and state-space greatly.

In addition to these axioms, each agent of ToE was supposed to have an influence on the game, and this control should be fairly distributed. However, in some situations playing ToE, individual, legal moves, can result in collective illegal moves. In the ToH game what defines an illegal move differs from a ToE game, since in the latter the direction-buttons do not influence the game, and the game is played alone.

During a ToH game, pressing the “GO” button with number of steps but not direction being chosen, results in a “No Direction” error message (see e-appendix chapter3\_45\_nodirection\_error). During a ToH game, trying to move a bigger disk onto a smaller disk results in a “Wrong Move” error message (see e-appendix chapter3\_46\_wrongmove\_error).

During a ToE game, pressing the “GO” button with number of steps but not direction being chosen, does not result in an error message. This is because such an error message can be regarded as deception. As the direction buttons do not influence the game, such an error message would state that the direction buttons have to be selected, implicitly saying that they actually do influence the game. Therefore, no such error message will pop up when no direction buttons are chosen. During a ToE game, when the resulting operator, consisting of three individual actions, would result in an illegal move, axiom number two would be violated. Following the meta-logic of the algorithm, such an operator will

be “corrected”. Ignoring the more complex meta-logic, the resulting solution is that a “M” disk will simply “follow its direction” until a rod is being reached, where it can be placed. This must not be the original rod, as this would violate axiom one. Therefore, there does not exist any error message for this case. This approach differs from the legacy zTree version, where in one instance such an error message was displayed to the ToE agents—however, this was due to a wrong implementation of the algorithm; another reason for discarding the legacy experiment.

In both ToH and ToE games, disks can jump edges. In other words, when the small disk is moved two steps to the right, the resulting state will always equal if the disk was moved one step to the left. For example, when starting the game (ToH and ToE) in state 1, an operator equal to “S2 right” will result in state 2 just as an operator equal to “S1 left”. No error message, for both ToH and ToE will popup, when a disk is moved “out-of-bounds”. In other words, there are “no borders” between the left rod and the right rod, as the graphical representation of the “Tower of Hanoi” experiment might suggest. This feature was introduced to make the ToH more similar to the ToE game, as the latter requires there to be “no borders” in order to make the algorithm work. A similar ruleset was used in the “Flag Run” game, as this experiments rests on the same meta-logic.

An example showing the correction of an illegal move during a ToE game is provided in the following. For instance, consider state 17 is reached and the goal rod was the center rod. The ideal ToH operator would be “M1 right” or “M2 left” applying “no borders” logic. During a ToE game, all three agents could use their single player logic and choose “M1, M1, M1”, which is “move\_state” 43, and results in the middle disk being moved one step to the left. The middle disk would jump edges from the left rod and land on the right rod, counting as “one step to the left”. This would result in an illegal state, as the small disks rests on the right rod. The medium sized disk therefore goes left one more step, landing on the center rod, where it may “legally” be positioned. This results in state 16, ultimately desired by the three agents. Therefore, the middle or large sized disks will always land on the only legal spot, when being chosen. This is not because the algorithm was designed in that way, but because the “playing field”, being just three rods, is so limited that it may look this way. The third axiom works in favor for the agents. This is because when all three agents agree on one disk being moved, such as “S<sub>x1</sub>, S<sub>x2</sub>, S<sub>x3</sub>” or “M<sub>x1</sub>, M<sub>x2</sub>, M<sub>x3</sub>”, the disk agreed upon will always be moved. If that chosen disk has to be selected in order to follow the most efficient path, and if that disk was a medium or large sized disk, it will always result in the ideal state, disregarding the choice of steps. This is the reason

why sticking to the ToH logic during a ToE game, will outperform randomness, as will be explained later.

Individual inputs can differ from the collective operator, as being shown in the following example. Consider state 14 being reached, the center rod being the goal rod. The ideal single player solution would be “S1 right” or “S2 left” applying the “no borders” logic. During a ToE game, all three agents could use their single player logic and choose “S1, S1, S1”, assuming no one had figured out that disks jump edges. This action input equals “move\_state” 1, and results in the small disk being moved one step to the left. The small disk would jump edges from the left rod and land on the right rod, counting as “one step to the left”. This would result in state 16, which was probably neither desired nor expected by the three agents.

It is clear by the two former examples that during a ToE game, several “logical” perspectives exist and that a ToE operator does not necessarily equal what an agent expected to happen. Whether or not an individual action equals some logic or whether or not the individual action results in some expected state is also expressed by raw data. The following sub-chapter will explain logic and expected state data.

#### 4.1.12 Logic and Expected States

The “Flag Run” experiment had shown that participants developed different strategies, all stemming from a different logic, but all effective strategies relied on having obtained “true rules” governing the game (Strunz & Chlupsa, 2019). During the “Flag Run” game, some participants even came up with effective strategies not anticipated by the experimenters; still, they also had to build their strategy upon having figured out a “true rule”. Such a “true rule” was that during the “Flag Run” game, the game-piece can jump edges or that the direction buttons did not influence the game at all. These two “true rules” are also implemented in the “Tower of Hanoi” experiment. During a ToH or ToE experiment, the disks can jump edges and the ToE game is not influenced by the direction buttons. Another “true rule” for ToE is that the disk, steps and direction are decided by all three agents as a collective, calculated by a complex algorithm, making it very unlikely or even impossible for one individual agent to control the game alone. However, participants can beat randomness by sticking to the ToH logic, as three actions that agree on a disk will result in this disk being moved.

Given the two examples from the former sub-chapter, it has already been described that different “logical” approaches can be identified, by which participants act. As the results of the “Flag Run” experiment had shown, most participants

who effectively solved the NRP stick with a single form of logic, as soon as feedback confirms this logic to be effective. Again, consider state 14 being reached, the center rod being the goal rod. The ideal single player solution would be “S1 right” or “S2 left” applying the “no borders” logic. When no agent had figured out during the ToH game that pieces “jump edges” and that there was “no borders” between the left and right rod, the “S2 left” solution might seem illogical. This is because the agent is only locally informed and is missing information about the true nature of the game, analogue to the island Gedankenexperiment. This might lead to all agents providing inputs “S1, S1, S1”, direction “right”, leading to state 16, which is unfavorable. However, it might just be that one agent had figured out the “no border” logic, applying “S2”, direction “left”. If the agents were able to inspect their co-agents’ inputs, the two remaining agents would find “S2”, direction “left” as being an illogical move. As a reminder, during a ToE game, the directions chosen by the players does not have any effect whatsoever. Assuming one agent chooses “S2”, and two agents choose “S1”, depending on the order of information, three different “move\_states” are possible: move\_state 2 (“S2, S1, S1”) moves the small disk one step to the right, leading to the desired goal state 15; move\_state 5 (“S1, S2, S1”) moves the small disk two steps to the right, leading to the unfavorable state 16; move\_state 17 (“S1, S1, S2”) moves the small disk one step to the right, leading to the desired goal state 15.

Assuming move\_state 2 and 17 occurred, could lead to a confirmation of a locally logical, but globally illogical strategy. The two agents who had chosen input “S1”, direction “right”, who had not figured out the true rule of “disks jumping edges” would find the input “S2”, direction “left”, as being an illogical input. From the perspective of the single agent, who used the “no border” strategy, both inputs are logical solutions. However, all logics are globally imperfect as the direction of disks cannot be influenced by pressing the direction buttons. This example shows the analogy to the island Gedankenexperiment. Depending on the individual experience, individual logics are applied, which can be either confirmed or denied by environmental conditions. However, since the agents decide collectively, negative feedback cannot be assigned to a wrong strategy with certainty. Group performance still can benefit from the “traditional strategy”, being acquired in the first ToH game, since agreement on the optimal disk outperforms randomness. While the rules of the game do not change, the goal rod’s position, simulating an environmental condition, can have a major influence on the strategy’s performance. A game-group can solve a ToE game optimally in 7 moves by applying the most intuitive “border” ToH logic, when the goal rod is the right rod (goal == 3). Since agents cannot communicate, even having obtained the excel sheet with all 64 input permutations, agents could not control the game fully, as



they had to communicate their input order. With the goal rod being the center rod, the most intuitive “border” ToH logic will fail to solve the game in 7 moves, and would lead to—probably—unfavorable and unexpected results. This environmental condition (goal == 2) will have an impact on any strategy’s performance. Depending on the game setup, feedback, valence weighting bias, routine strength, logic applied and intrinsic motives, a human agent will alter its strategy or remain using the strategy. It is therefore important to cover as many “logics” as possible, in order to make sense of the agent’s strategy, and to measure if an input lead to an “expected” output state.

All different logic models and “expected” output states are saved as a Boolean in ToH.csv and ToE.csv (see e-appendix chapter3\_dir\_or\_nodir\_states\_appendix\_47 and chapter3\_exp\_states\_appendix\_48). With their necessity being explained, the following will explain which models of logic are saved, how they are created and how “expected” and “unexpected” states are distinguished.

During a single player ToH game, two different “logic models” are being measured. These two logic models are saved in ToH.csv as a Boolean with “logic” and “no\_border\_logic”. When the Boolean of the according “logic model” equals 1, the action is regarded as being equal to this “logic model”. When the Boolean equals 0, the action is regarded as not being equal to this “logic model”.

In ToH an action is saved as being equal to “logic”, with Boolean equal to 1, when the output state follows the ideal path, without the disk “jumping edges”. This path depends on “start” and “goal” rod position, “start\_state”, “input” and “direction”. In ToH an action is saved as being equal to “no\_border\_logic”, with Boolean equal to 1, when the output state follows the ideal path, with the disk “jumping edges”. The ideal path is the path were the goal is being reached in 7 moves. When the playing piece deviates from the ideal path, both “logic models” assign a Boolean of 0. When the “start\_state” is a state, which is the result of such a deviation, the following move is given a Boolean of 1, when it follows the ideal path again. The latter is to be shown by example in the following. All possible configurations that lead to “logic” or “no\_border\_logic” are listed in the electronic appendix (see e-appendix chapter3\_dir\_or\_nodir\_states\_appendix\_47).

For instance, consider a ToH game with the starting rod being the left rod, the goal rod being the right rod and three disks. In this case, the ideal path would lead to “state” 2, and the ideal operator for “logic == 1” would be “S2 right”, and the ideal operator for “no\_border\_logic == 1” would be “S1 left”. Studies about “Tower of Hanoi” performance showed that the first move is often the move, deviating from the ideal path the most. Assuming the agent deviates from the ideal path at the first move, choosing action “S1 right”. State 9 is reached and Booleans for both “logic models” are then assigned a value of 0. The ideal path

would now lead to “state” 2, and the ideal operator for “logic == 1” would be “S1 right”. Assuming the agent chooses to correct its deviation by “S1 right”, state 2 is reached and a Boolean of 1 is assigned to “logic”, even though the “start\_state” was outside of the original ideal path. A Boolean of 0 is assigned to “no\_border\_logic”, since the disk did not jump edges.

During a three player ToE game, where the direction buttons do not influence the disks, several different “logic models” are being measured. Players can also choose to confirm an input without having pressed a direction button—being saved as “n” or “none”. ToE logic models are saved in ToE.csv with a Boolean assigned to them. When the Boolean of the according “logic model” equals 1, the action is regarded as being equal to this “logic model”. When the Boolean equals 0, the action is regarded as not being equal to this “logic model”.

The ToE “logic models” are: “framed\_logic”, distinguished by “dir”, “no-dir” and “ideal”; “no\_border”, distinguished by “dir”, “nodir”, and “ideal”. Each “logic model” evaluates individual actions, and not the collective operator. Written in front of each “logic model” attribute, “first\_player”, “second\_player” or “third\_player” refer to which agent’s action is considered.

The logic model “frame\_logic” with “dir”, short for “direction”, is isomorphic to ToH “logic”. When participants play one or multiple games of ToH, routine strength by feedback leads to human agents probably “carrying” the single player ToH logic into the three player ToE domain.

A Boolean of 1 is assigned to “framed\_logic” with “nodir”, short for “no direction”, when an action equals “framed\_logic\_dir”, disregarding direction. For example, consider a ToE game with the starting rod being the left rod, the goal rod being the right rod and three disks. In this case, the ideal path would lead to “state” 2, and the action for “framed\_logic\_dir == 1” would be “S2 right”, and the action for “framed\_logic\_nodir == 1” would be “S2 left” or “S2 right” or “S2 none”. This logic is measured because during the legacy experiment, participants had orally reported to the experimenter that they kept on pressing direction buttons, even though they were informed that they did not influence the game. They remained pressing the direction buttons arbitrarily, either always or frequently, without using “framed\_logic\_dir”. They reported to do so to “stay in rhythm” or “out of routine” or “because they did not feel like ignoring the direction buttons altogether”.

A Boolean of 1 is assigned to “framed\_logic” with “ideal”, when an action equals “framed\_logic\_dir”, and direction “none” is chosen. For example, consider a ToE game with the starting rod being the left rod, the goal rod being the right rod and three disks. In this case, the ideal path would lead to “state” 2, and the action for “framed\_logic\_dir == 1” would be “S2 right”, and the action

for “framed\_logic\_ideal == 1” would be “S2 none”. This logic is measured to identify players, who stick with the “framed\_logic” but regarded the direction buttons as being useless.

The three remaining “logic models” are analogous to the three mentioned “logic models”, however, they identify players who make the disk “jump edges”. All three versions of “no\_border logic models” were introduced to identify players, who had obtained the “true rule” that “disks jump edges”.

Assuming a ToE game with the starting rod being the left rod, the goal rod being the right rod and three disks. In this case, the ideal path would lead to “state” 2, and the action for “no\_border\_dir == 1” would be “S1 left”, the action for “no\_border\_nodir == 1” would be “S1 left” or “S1 right” or “S1 none”, and the action for “no\_border\_ideal == 1” would be “S1 none”.

The following Table 4.4 (own source) will list all mentioned examples of ToE “logic models”.

**Table 4.4** All possible input combinations resulting in according logic category booleans

ToE, with starting rod being left rod, goal rod being right rod, first move		
Individual input	Logic model	Boolean
S2 right	framed_logic_dir/nodir/ideal no_border_dir/nodir/ideal	1/1/0 0/0/0
S2 left	framed_logic_dir/nodir/ideal no_border_dir/nodir/ideal	0/1/0 0/0/0
S2 none	framed_logic_dir/nodir/ideal no_border_dir/nodir/ideal	0/1/1 0/0/0
S1 right	framed_logic_dir/nodir/ideal no_border_dir/nodir/ideal	0/0/0 0/1/0
S1 left	framed_logic_dir/nodir/ideal no_border_dir/nodir/ideal	0/0/0 1/1/0
S1 none	framed_logic_dir/nodir/ideal no_border_dir/nodir/ideal	0/0/0 0/1/1

With models of logic explained, i.e. how they are created, saved and for which purpose they were introduced, “expected” and “unexpected” states are to be discussed in the following.

In a ToE game unexpected outputs can occur. For this reason, ToE.csv assigns a Boolean to each individual agent’s action, depending on the output state, indicating whether (1) or not (0) the individual action lead to the expected outcome.

This measurement was introduced to obtain data about individual decision-making correlating with feedback or in other words, to obtain information about whether or not participants alter their strategy when they “do not see what they expected” or stick to a strategy when they “do see what they expected”. In order to make an assumption about, whether or not some output state was expected by a participant, several data attributes have to be known: start\_state, output state, input, and direction. A data sheet lists all possible configurations (see e-appendix chapter3\_exp\_states\_appendix\_48).

Several “expectation models” exist, with each assigned a Boolean: FL\_exp\_dir, short for “framed logic expectation considering direction”; FL\_exp\_ideal, short for “framed logic expectation not considering direction”; NB\_exp\_dir, short for “no borders logic expectation considering direction” and NB\_exp\_ideal, short for “no border logic expectation not considering direction”.

For instance, consider a ToE game with the starting state being equal to state 1, as modelled in the state-space (see Figure 4.4). Some game-group operator results in the output state equal to state 2, as modelled in the state-space. An input equal to “S2 right” leads to Boolean 1 being assigned to “FL\_exp\_dir”. An input equal to “S2 none” leads to Boolean 1 being assigned to “FL\_exp\_ideal”. Both “expectation models” assume the “framed logic model”. When the “no border logical model” is assumed, two other “expectation models” are distinguished. Considering the same example start and output state by some game-group operator, an input equal to “S1 left” leads to Boolean 1 being assigned to “NB\_exp\_dir”. An input equal to “S1 none” leads to Boolean 1 being assigned to “NB\_exp\_ideal”.

How logic models and expectation models are to be interpreted depends on many factors. To reduce complexity, a heuristic approach is being taken: it is assumed that every input is being chosen deliberately. Therefore, with “expected models”, there does not exist an “exp\_nodir” distinction. This, of course, excludes errors and deviations stemming from misclicking or non-deliberate inputs. It also makes data of “expectation models” meaningless, when the participant expected “nothing” or just “randomly” provided inputs. It is assumed such deviations occur rare enough, so that their number have a neglectable impact on the overall experiment. However, when a participant shows many “framed\_logic\_nodir” actions, and next to no “dir” or “ideal” data, it can be assumed that data of “expectation models” will not be very meaningful, as they assume the direction to be selected with purpose.

Building upon Rubinstein (2007) an action can be considered “reasonless”, when response times are short, and also by analyzing the actions’ logic and expected states, i.e. a set of actions that jump between different logic states, and has an expected state outcome similar to the randomizer bot results, can efficiently

regarded as being “reasonless”. An action can be considered “instinctive” when response time are short, and e.g. both logic and expected states show that the agent still sticks with the single-player logic routine, being “framed” by its own mental model. While Rubinstein (2007) categorized actions intuitively, this thesis follows Rubinstein’s suggestion and base the categorization of actions between cognitive, instinctive and reasonless with “on other sources of information”, being the logic and expected states data (Rubinstein, 2007, p. 1258). This also reduces the risk to falsely interpret agent deviations from optimal behavior, e.g. by simple misklicking as non-standard preferences (Cason & Plott, 2014).

This concludes all logic and expected states models. With all raw data and their creation explained, the next sub-chapter will focus on the participants, who conducted the main experiment.

---

## 4.2 Participants

180 Amazon Mechanical Turk workers (MTurk) were recruited via “Amazon Mechanical Turk”, where online freelancers can be hired for various tasks, such as online questionnaires and experiments. From these 180 participants, data of 87 MTurks could be used for statistical analysis. As indicated in Strunz & Chlupsa (2019), MTurks “are commonly recruited for behavioral experiments due to AMT’s workers pool size, low costs and being able to produce high-quality data fast (Buhrmester, Kwang, & Gosling, 2011)“ (p. 114). Freelancers recruited via this platform show comparable bias and heuristic behavior as participants recruited by more traditional methods (Paolacci, Chandler, & Ipeirotis, 2010). MTurks are mainly motivated monetary compensation (Lovett et al., 2018), such that realistic working conditions can be simulated with these participants, where thinking-time is associated with costs.

There exist possible cultural influence on complex problem solving and adaptive decision making (Güss, 2011; Güss et al., 2012)—differences which are supposed to stem from different learning environments (Funke, 2014). Highly significant differences in non-routine problem solving performance and response times by country origin have been measured comparing 290 Indian, 262 US-American and 51 German participants via the “Flag Run” experiment (Strunz, 2019). For this reason, all 180 MTurks were restricted to US American MTurks. In order to ensure that MTurks were actually human and not automatically working machines, so called “bots”, approval rating, reflecting the MTurk’s „reputation“ was set to “high levels” to ensure high quality data. High levels of MTurk reputation are defined to be the case with an approval rating above 95% (Peer, Vosgerau,

& Acquisti, 2013). When a task, referred to as “HIT” is opened to US American freelancers with a mandatory HIT approval rate of higher than 95%, 11,126 freelancers were “captured” in a study from 2015 (Stewart et al., 2015) and a more recent study stated there being 12,000 MTurk freelancers on average (Difallah, Filatova, & Ipeirotis, 2018). However, according to Difallah, Filatova & Ipeirotis (2018), these numbers are extreme underestimates due to variation. When correcting for propensities at least 100,000 to 200,000 freelancers are actively working.

Even though no differences in NPS performance were measured regarding sex, and only low correlation regarding age were found in Strunz & Chlupsa (2019), age and sex was again asked for during the login-stage, as the reflective cognitive state was described as being influenced by age (Liebherr, Schiebener, Averbeck, & Brand, 2017) and as female participants have shown to change to a better strategy less efficiently in experiments under feedback (Casal, DellaValle, Mittone, & Soraperra, 2017).

As MTurks’ behavior vary over the course of a 24hour day, with participants behaving less reflective on the weekends (Arechar, Kraft-Todd, & Rand, 2017), the final study was conducted on a regular working day, being the 6<sup>th</sup> of December. According to an online tracker showing hourly demographics of AMT Workers (Difallah et al., 2018; Paolacci et al., 2010), the most recent data available at the time when the experiment was conducted, showed variation of US American freelancers throughout the entire month of September 2019 ranging between 52.58% and 85.42%. The the majority of MTurks consisted of US Americans. Regarding sex, 49.04% female and 50.96% male US American MTurks participated from September 1<sup>st</sup> 2019 to September 30<sup>th</sup> 2019 according to the online tracker (Paolacci et al., 2010), indicating well balanced monthly sex distribution. Dates for December, when the experiment was conducted, were not available at the time of research.

Age distribution over workdays, being Monday to Friday, from the 1<sup>st</sup> of September 2019 to 30<sup>th</sup> of September 2019 retrieved from the online tracking tool (Paolacci et al., 2010) are listed in Table 4.5.

MTurk demographics from 2018 (Difallah et al., 2018) reported 55% of US female participants and 45% of US male participants. Household income for US MTurks were found to be below the average of the US population: with the US household median being “\$57K” and the US MTurks household median being “around \$47K”, and “while 26.5% of US households make more than \$100K per year, for MTurk workers this percentage falls at 12.5%.” (Difallah et al., 2018, p. 4).

**Table 4.5** Year of Birth distribution of MTurks on workdays Mo-Fr, from 01.09.2019 to 30.09.2019. *Source* data acquired via online tracking tool by Paolacci et al., 2010

Year of Birth	Percentage	Age (as of 10/2019)
2000–2010	1,268%	9–18
1990–2000	34,232%	19–28
1980–1990	36,87%	29–38
1970–1980	14,758%	39–48
1960–1970	9,46%	49–58
1950–1960	2,604%	59–68
1940–1950	0,598%	69–78
1910–1940	0,21%	79–108

From 87 MTurks 29 self-reported being female and 58 self-reported being male, with an average of 33.16 years for both sexes.

Actively monitoring the MTurk forums is recommended by researchers, in order to find out whether or not a HIT was discussed amongst the MTurks, which could have a negative influence on the experiment's data quality (Cheung, Burns, Sinclair, & Sliter, 2017). When a pre-test of the main experiment was performed, minor information about the experiment was found to be shared online. A single participant rated the experiment as “fair” but also stated the disadvantage that one of his partner's bad performance made her wait longer than necessary. Information being shared online cannot be avoided. For this reason, an after-survey was included, asking participants, whether or not they had already participated in this experiment before.

As most participants are informed about playing in a group anyways, information being shared online was observed to be very limited, and as certainly not all MTurks are actively monitoring the MTurk forums, treatment diffusion effects are regarded as potentially low. For this reason, more transparency was regarded to outweigh its potential negative side-effects, and an official profile on “TurkerView” was created, where MTurks are able to retrieve information about the experimenter's former payments, communication, number of rejections, approval response times, and number of blocked participants.

As studies have found 40% of MTurks working with “Amazon Mechanical Turk” as their primary job, the practical recommendation to act as “reputable employers” was followed (Brawley & Pury, 2016, p. 542), and more than 45 USD per hour was paid to MTurks on average over the course of 26 HITs. Since “unfair wages, and inaccurately listed time requirements were among the top five worst Requester behaviors” (Brawley & Pury, 2016, p. 542), calculating MTurks

average pay was always aimed way above US minimum wage, when experience in early experiments was missing. For this reason, the high average hourly pay was achieved.

In conclusion, from 180 participants, data of 87 US American MTurks was randomly selected from an online pool of potential participants. How many MTurks can ultimately be reached is debated and dependent on the model used to approximate it. According to literature, certainly more than 11-thousand freelancers via “Amazon Mechanical Turk” were reached, and numbers could extent to more than 100-thousand. In order to being able to rely on the most recent statistical results, data from September 2019 was used to determine US female/male distribution, and age during working days.

---

### 4.3 Procedure

Participants were provided with in-depth instructions and a text field, where the according participants ID was supposed to be entered, as shown in Figure 4.4.



READ Instructions HERE, 2-3 Minutest (Click to expand)

Survey link:

Provide the survey code here:

Submit

**Figure 4.4** MTurk client side view of HIT. *Source* own source

Upon having clicked on “(Click to expand)” each participant was provided with the following instructions:

**Survey Instructions reading time: 2–3 minutes. Trouble Shooting section included. Make sure to read.**

Complete an online experiment consisting of 12 levels of Tower of Hanoi. Instructions are included ingame. Bonus pay for best 10%.

= When experiment lasts longer than 51 minutes, submit with Worker ID and time played, you will be approved if you did not idle on purpose. =



Go to <https://www.curiosity-data.com/> and enter **1992** as “Session Code”.

Please provide us with your sex and age. You will be given an ID at the end. Enter this ID as your Surveycode. Do **not** provide me with your worker ID.

This experiment may easily last longer than 30 minutes. Do not start this HIT when you do not have enough time.

You may have to **wait up to 10 min** at the beginning.

You may have to **wait up to 14 min** during the experiment.

Do not leave the game unattended. If you are kicked due to inactivity, I will **under no circumstances** approve your work.

Check the information box on the left of your screen during the game. Its contents may change and are important.

**Make sure to leave this window open as you complete the survey.** When you are finished, you will return to this page to paste the ID into the box. **Not** your worker ID.

**About me:**

I am registered on TurkerView (Ulrich Strunz), if you want to leave a rating.

You can easily reach me via email, I will answer.

My experiments are unique. Thanks for helping me out.

**Compensation:**

In case you are unable to submit in time, I offer compensation in some cases. My time is limited. I am also human. Please be patient in this case, I am working with hundreds of MTurks simultaneously, alone. Leave me a reminder Email in case you did not receive funds. Screenshots help, so you can prove your progress.

= When experiment lasts longer than 51 minutes, submit with Worker ID and time played, you will be approved if you did not idle on purpose. =

Please do not spam me with multiple emails, I will listen to your explanations in case something went wrong.

**Survey:**

There is a pre-survey included. Please make sure to answer it. I need to know if you are from USA.

There is an after-survey included. Please make sure to answer it. I need to know if you have played this experiment before.

**Experiment:**

Using a tablet or notebook will be ideal. Mobile phones might have a too small screen to display all information properly.

The experiment is not bugged. It has been tested with more than 200 participants by now. I have no influence over the setup you are using. Old hardware or missing drivers may result in bad latency. Check the trouble shooting section for more details.

**Trouble-Shooting:**

**!!! Some MTurks experience problems when using Google Chrome since its last update. Clearing your cache might be necessary before starting the game. !!!**

The game has been tested with Chrome, IE, Firefox, Yandex Browsers. No trouble was experienced.

In case you accidentally close your browser, just come back. Your experiment progress will be saved.

Several Turkers reported a problem with the publish button not working. This is an AMT specific problem. The best option is to:

- 1) Inform me via Email when an error occurs.
- 2) Wait. Sometimes the button will function after 5 min of waiting time.

In case the proceed button does not work:

- 1) Make sure you have typed in **1992** as session code.
- 2) Make sure to have chosen your SEX using the drop-down menu and have provided us with your AGE using integers.
- 3) Since the latest Chrome update, some unsolvable (from my side) issues were reported, when using this Web-Browser.

After going to <https://www.curiosity-data.com/>, participants had to self-report their sex and age. After a valid input they could start the experiment by clicking the “Proceed” button. The entire experiment consisted of a Tower of Hanoi/Tower of Europe experiment, and an after-survey. The experiment came with five different information conditions, represented by five experiment groups. The participant was assigned to one of the five experiment groups automatically by login order, as explained in the former chapter. Participants were also assigned automatically to game groups by login order, each game group consisting of three players, as explained in the former chapter. Each information condition (experiment group) consisted of 13 games or levels. The first 7 games were single player games. The last 6 games were multiplayer games. The first game was

added to give players enough time to read popup and help-text information, and data collected during the first game was not used for analysis. The first game was considered as a buffer level. The second popup showed up at game 8. The goal rod changed during the single player and multiplayer games. Each game was played with three rods, referred to as either left, center or right rod. During the first four single player games (buffer level included) the goal rod was set to be the right rod. During the last three single player games, the goal rod was set to be the center rod. The first three multiplayer games were played with the right rod being the goal rod, while the last three multiplayer games set the goal rod to be the center rod. All games in all information conditions were played with three disks. Figure 4.5 (own source) shows the entire setup.

For each single player game, a timer of 2 minutes was preset. When the timer ran out a level was automatically ended, skipping to a “Level Completed” screen, which was also automatically closed after 1 second, having shown the next game screen. The single player timer automatically started as soon as the game screen was shown.

For each multiplayer game a timer of 3 minutes was preset. When the timer ran out a level was automatically ended, skipping to a “Level Completed” screen, which was also automatically closed after 1 second, having shown the next game screen. If the last game 13 was skipped in such a way, the after-survey was automatically displayed.

When a game group was not filled with three participants before the experiment started, a waiting screen appeared. After 5 min of timeframe an automated bot participant was added to the group. If for some reason a player left a game group during the game, and Curiosity IO registered this player as being disconnected, a bot was added to the group after 10 min. This feature was implemented for ethical reasons, so that MTurks were still able to solve the experiment in time. When a game group was filled with three agents, the actual experiment started with the first game level.














The first level called “Game 0” includes a popup with the following message:

**“--- Do not worry about the timer. Take your time to read the following! ---**

Your task is to solve 12 games of Tower of Hanoi.

6 training games, and then 6 performance games. An additional game (this game) is added, so you can read these instructions.

For each game a timer will be displayed. When the timer reaches zero, the next game will automatically start.

Game	Discs	Start	Goal	Type	HelpText	Popup	
Game 1	3	1	3	single	Yes	Yes	
Game 2	3	1	3	single	Yes	No	
Game 3	3	1	3	single	Yes	No	
Game 4	3	1	3	single	Yes	No	
Game 5	3	1	2	single	Yes	No	
Game 6	3	1	2	single	Yes	No	
Game 7	3	1	2	single	Yes	No	
Game 8	3	1	3	multi	Yes	Yes	
Game 9	3	1	3	multi	Yes	No	
Game 10	3	1	3	multi	Yes	No	
Game 11	3	1	2	multi	Yes	No	
Game 12	3	1	2	multi	Yes	No	
Game 13	3	1	2	multi	Yes	No	

**Figure 4.5** Administrator perspective of entire experimental setup using Curiosity IO framework. *Source* own source

Try to solve each level in as few steps as possible. The best 10% of all participants will win a 2.00 USD bonus (only if you provide me with the ID displayed at the end of the experiment, do NOT provide me with your worker ID).

Your performance will not be measured during the first 6 practice games.

Your performance will be measured during the 6 performance games.

Important:

1) During performance games, the timer will start AFTER your first move. So you can take your time reading pop-up information.

2) Pay close attention to the instructions on the left-hand side as they might change. A pop-up will be displayed when additional information is added to the instructions, to make sure you notice the change.

3) Every piece of information displayed is true. You can trust all written information.”

Instructions displayed on the left side included the following text:

**“Instruction Game 0**

(no performance measured)

The objective of the puzzle is to move the entire stack to the indicated goal rod, either center or right rod, obeying the following simple rules:

Only one disk can be moved at a time.

Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack or on an empty rod.

No larger disk may be placed on top of a smaller disk.

Click on the disk you want to move first. Drag-and-Drop does not work. After that, you will have to figure out the rest for yourself.

With 3 disks, the puzzle can be solved in 7 moves.

Additional information:

No additional information so far.”

The instruction text did not change throughout the first 7 games, except in the integer referring to the current level being played.

A second popup appeared in all five conditions with starting of game 8, and instruction texts differed amongst conditions. All instruction texts were altered as follows:

*“Performance Game 7 (performance is measured)”*,

therefore, participants were informed about that their performance was now evaluated during the coming levels.

Instructions texts then differed amongst the five conditions (experiment groups, EG) in the “Additional Information:” part. The second popups differed from the first popup by stating “You are now starting with 6 performance games.” and with exception of experiment group 1, the second popup also contained the warning phrase “Attention! Additional information was added to the instructions. Make sure to read!”. This warning was implemented in order to make sure that participants actually read the additional information. The additional information contents were written in capital letters to induce disfluency, in order to enhance chances of “promoting more comprehensive consideration of opposing views”, as disfluency in writing style has been proven to disrupt confirmation bias (Hernandez & Preston, 2013, p. 178).

Instruction text content is summarized in the following Table 4.6.

**Table 4.6** Instruction texts for according information conditions in „ill-defined“ stages.  
*Source* own source

EG	Game 8 instruction text “Additional Information:”	Warning
1	“No additional information.”	No
2	“YOU ARE PLAYING IN A TEAM OF THREE HUMANS DURING THE NEXT 6 GAMES. YOU ALL HAVE INFLUENCE ON THE MOVEMENT OF THE DISCS AND SHARE CONTROL OVER THE GAME ACCORDING TO HIDDEN RULES. THE RULES DO NOT CHANGE DURING THE NEXT 6 GAMES.”	Yes
3	“YOU ARE PLAYING IN A TEAM OF THREE HUMANS DURING THE NEXT 6 GAMES. YOU ALL HAVE INFLUENCE ON THE MOVEMENT OF THE DISCS AND SHARE CONTROL OVER THE GAME ACCORDING TO HIDDEN RULES. THE RULES DO NOT CHANGE DURING THE NEXT 6 GAMES. SINCE YOU CANNOT COMMUNICATE WITH EACH OTHER, IT IS HIGHLY UNLIKELY FOR YOU TO FIND OUT THESE RULES.”	Yes
4	“YOU ARE PLAYING IN A TEAM OF THREE HUMANS DURING THE NEXT 6 GAMES. YOU ALL HAVE INFLUENCE ON THE MOVEMENT OF THE DISCS AND SHARE CONTROL OVER THE GAME ACCORDING TO HIDDEN RULES. THE RULES DO NOT CHANGE DURING THE NEXT 6 GAMES. DURING THE NEXT 6 GAMES THE DIRECTIONAL BUTTONS DO NOT INFLUENCE THE GAME AT ALL. ALL THEY DO IS CHANGE COLOR WHEN BEING PRESSED.”	Yes
5	“YOU ARE PLAYING IN A TEAM OF THREE HUMANS DURING THE NEXT 6 GAMES. YOU ALL HAVE INFLUENCE ON THE MOVEMENT OF THE DISCS AND SHARE CONTROL OVER THE GAME ACCORDING TO HIDDEN RULES. THE RULES DO NOT CHANGE DURING THE NEXT 6 GAMES. SINCE YOU CANNOT COMMUNICATE WITH EACH OTHER, IT IS HIGHLY UNLIKELY FOR YOU TO FIND OUT THESE RULES. DURING THE NEXT 6 GAMES THE DIRECTIONAL BUTTONS DO NOT INFLUENCE THE GAME AT ALL. ALL THEY DO IS CHANGE COLOR WHEN BEING PRESSED.”	Yes

With game level 8 being reached, participants played 6 rounds of Tower of Europe, in identical manner as explained in the former chapter. Each experiment group contained different additional information, defining the five different information conditions, which are to be explained in the following.

The first information condition (EG: 1) did not contain any further information. Participants were not informed about the fact that they did now share control

with two additional agents. This information condition is now referred to as “no information condition” (N-IC).

The second information condition (EG: 2) informed participants about them sharing control during all 6 performance games with two other agents in accordance to hidden rules, which will not change. This information condition is now referred to as “GDM information condition” (G-IC).

The third information condition (EG: 3) informed participants about them sharing control during all 6 performance games with two other agents in accordance to hidden rules, which will not change. Participants also received the “discouraging” information that due to a lack of communication potential these hidden rules will likely remain hidden. This information condition is now referred to as “disillusioning information condition” (D-IC).

The fourth information condition (EG: 4) informed participants about them sharing control during all 6 performance games with two other agents in accordance to hidden rules, which will not change. Participants also received the information about the directional buttons not having any function besides changing color when being pressed. This information condition is now referred to as “routine information condition” (R-IC).

The fifth information condition (EG: 5) contained all additional information from G-IC, D-IC, R-IC. This information condition is now referred to as “combined information condition” (C-IC).

Additional information content was displayed throughout all ToE games, and did not disappear or alter its contents at any moment.

Having solved all 6 ToE games, participants had to answer an after-survey, simply asking “Have you done the experiment “Flag Run” before?”, which participants were able to answer by either choosing “Yes” or “No”, after which the experiment ended, and participants were provided with their ID.

The following chapter will derive hypotheses, list dependent and independent variables and how data was treated.

**Open Access** This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.

