Towards Swift and Accurate Collusion Detection

Jouni Smed Timo Knuutila Harri Hakonen

Department of Information Technology
FI-20014 University of Turku, Finland
{jouni.smed, timo.knuutila, harri.hakonen}@utu.fi

ABSTRACT

Collusion is covert co-operation between participants of a game. It poses serious technical, game design, and communal problems to multiplayer games that do not allow the players to share knowledge or resources with other players. In this paper, we review different types of collusion and introduce two measures for collusion detection. We also propose a model and a simple game, implemented in a testbench, for studying collusion detection.

INTRODUCTION

When the rules of a game forbid the players to co-operate, any attempt of covert co-operation is called *collusion*. The players who are colluding (i.e., whose goal is to win together or to help one another to win) are called *colluders*. Collusion poses a major threat to games that assume that the players aim at individual goals individually, because many types of collusion are impossible to prevent in real time. Even detecting collusion can require discerning and understanding the player's motivation – which is often an impossible task for human beings, too. For this reason, collusion is usually detected only afterwards by studying the behaviour of the players and recognizing characteristic patterns that indicate forbidden co-operation.

Apart from games suspectible to collusion such as poker [2, 8, 10] and bridge [11], collusion have been addressed also in the context of tournaments [3] and multiple choice examinations [1]. In our previous work [7] we introduced a classification for different types of collusion, which we will present in the next section. We argued that different types of attacks have been lumped together under the same collective title "collusion" and that they have been commonly dismissed as unsolvable in the literature. We showed that although there are collusion types that are indeed impossible or very hard to detect, there are also cases where automatic recognition is possible. In this paper, we take one step further and present a model and a simple game with which collusion detection methods can be tested and evaluated. Our motivation is that only when we understand how to detect collusion, we can proceed further to its prevention.

The plan of this paper is as follows: We begin by presenting classifications for collusion. They line out the types of collusion and give us the terminology that we will use throughout this paper. After that, we look at the problem statement of collusion detection. It gives us measures upon which the models and testbench game presented next will rely. Finally, we will conclude the paper with a discussion on how the model presented in this paper helps the research and what are the steps for future work.

CLASSIFYING COLLUSION

When the players of a game decide to collude, they make a agreement on the terms of collusion [7]. This agreement has four components:

Consent How do the players agree on collusion?

- *Express collusion*: The colluders make an explicit hidden agreement on co-operation before or during the game.
- *Tacit collusion*: The colluders have made no agreement but act towards a mutually beneficial goal (e.g., try to force the weakest player out of the game).

Scope What areas of the game the collusion affects?

- *Total collusion*: The colluders co-operate on all areas of the game.
- *Partial collusion*: The colluders co-operate only on certain areas and compete on others (e.g., sharing resource pools but competing elsewhere).

Duration When does the collusion begin and end?

- *Enduring*: Collusion agreement lasts for the duration of the game.
- *Opportunistic*: Collusion agreements are formed, disbanded, and altered continuously.

Content What is being exchanged, traded, or donated in the collusion?

- *Knowledge*: The colluders share expertise (e.g., inside information on the game mechanics), in-game information (e.g., the colluders inform one another the whereabouts of the non-colluding players) or stance (e.g., the colluders agree on playing "softly" against one another).
- *Resources*: The colluders share in-game resources (e.g., donating digital assets to one another) or extra-game resources (e.g., a sweatshop is playing a character which will be sold later for real-world money).

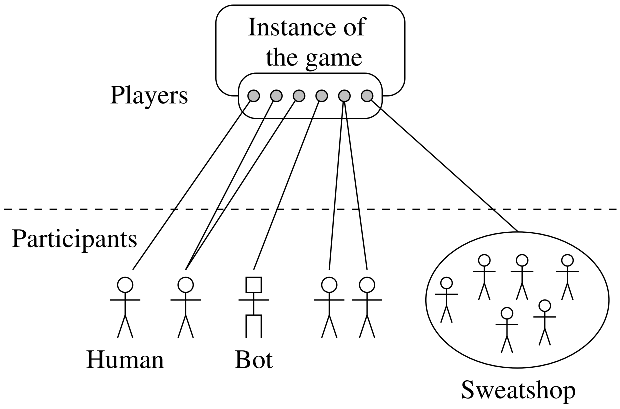

Figure 1: Players and participants are the partakers of a game. The relationship is usually assumed to be one-to-one, but one human participant can control two or more players, a player can be controlled by a computer program (i.e., a bot), or two or more participants (e.g., a sweatshop).

This classification is not sufficient for on-line computer games, because we must also discern the roles of the partakers – players and participants – of the game [7]. A player in a game can be controlled by one or more participants, and a participant can control one or more players in a game (see Figure 1). This means that there are two types of collusion: (i) collusion among the *players* which happens *inside* the game, and (ii) collusion among the *participants* which happens *outside* the game. To detect player collusion, we have to analyse whether the players' behaviour diverges from what is reasonably expectable. To detect participant collusion, we have to analyse the participants behind the players to detect whether they are colluding.

This gives a fine-grained classification of collusion types:

Participant identity collusion How a single player is perceived to participate in a game?

- *Player controller collusion*: Many participants are controlling one player (e.g. two players controlling the same character alternatively).
- *Self-collusion*: One participant is controlling many players (e.g. one participant controls many player in a poker table).

Inter-player collusion How the participants are affecting the game?

- *Spectator collusion*: Co-colluder provides a different type of information (e.g., ghost scouting, post-game information).
- *Assistant collusion*: Co-colluder plays (possibly sacrificingly) to assist the other to win (e.g., as a sidekick, passive scout, or spy).
- *Association collusion*: Colluders achieve individual goals through co-operation.

Game instance collusion How factors outside the game instance affect the game?

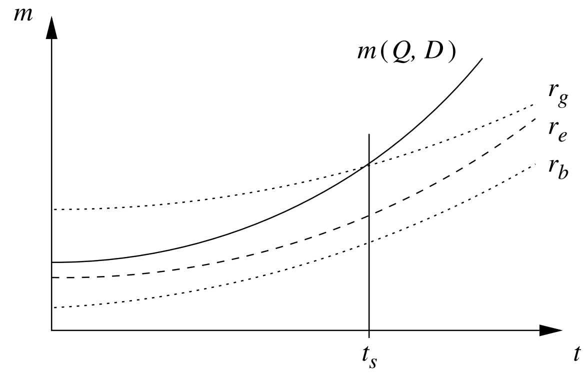

Figure 2: Collusion is detected when the observed results using a measure m deviate significantly from the expected results r_e. Suspicion arises at the moment t_s when the results are getting either too "good" and cross the threshold r_g or they are getting too "bad" and cross the threshold r_b.

- *Multigame collusion*: Players of different game instances collude (e.g., studying the game properties, finding suitable server, fixing tournament match results).
- *Insider collusion*: The co-colluder is a game administrator or game developer that reveals or modifies the workings of the game instance.

Because collusion prevention requires that collusion gets first detected, let us next take a closer look at what is required from collusion detection.

COLLUSION DETECTION

When comparing collusion detection methods, we should observe the following two properties:

Accuracy How justified is the suspicion raised by the detection method?

Swiftness How early does the suspicion raise?

Naturally, accuracy is important so that normal behaviour does not set off an alarm and cause uncalled for inspection or unjust punishment. Swiftness is usually related to accuracy so that the less accurate the detection is, the swifter the suspicion is detected.

Let us try to interpret these properties in a somewhat more formal – but simple – manner (see Figure 2). Suppose that our detection is based on applying some numeric function m upon the participants P of the game and some collected game data D. Let $Q \subseteq P$ and r_g is some chosen threshold value of the best possible play. If $m(Q,D) > r_g$, we decide that the players in the set Q are colluding. In this framework the questions to be asked are:

Accuracy How is the value of m related to the probability that Q really contains colluders?

Swiftness How much data D is needed before r_g is exceeded?

Ideally, we would like to have a measure that indicates as early as possible when players are colluding or when their

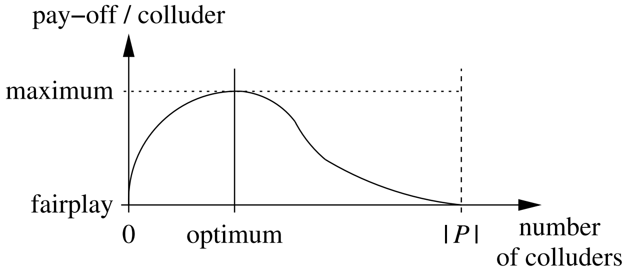

Figure 3: The pay-off of collusion per colluder increases until the optimum number of colluders is reached, after which it approaches asymptotically fairplay pay-off.

behaviour is showing suspicious traits. Should the detection happen before collusion actually gives any notable gain for the colluders, we have managed to prevent it altogether. How then to find such methods? From an intuitive point of view, any abnormal behaviour in a game should raise a suspicion. This is the case especially when some of the players get too good (i.e., exceeding the threshold r_g) or too bad (i.e., going under the threshold r_b) results in comparison to their playing skills (the latter would indicate a case of assistant collusion). Function m could then indicate the (absolute) difference between the expected behaviour (e.g., wins in a card game) against the observed one.

How to select Q then? Instead of inspecting all $|\wp(P)| - |P|$ different colluder sets, we can limit $|Q|$ to a certain range, which depends on the collusion pay-off of the game. Figure 3 illustrates the pay-off of collusion with respect to the number of colluders. As the number of colluders increases, the total amount of pay-off also increases. However, when the pay-off is divided among the colluders, there exists an optimum where the pay-off per colluder is the greatest. For example, robbing is more effective when the gang of robbers is big, but a big gang of robbers has to focus on big heists to provide everyone with a big share of the loot. When we are detecting colluders, $|Q|$ can be limited near to this optimum. For the game design this means that it is possible to discourage large-scale collusion by pushing down the peak of the curve. For example, if robbing is allowed in the game but a part of the loot gets damaged (or otherwise loses its value), the optimum size of a gang of robbers gets smaller.

Next, let us limit ourselves to inter-player collusion, where the players of the game co-operate by exchanging in-game resources or information. This type of collusion is what is "normally" understood as collusion, where we can assume players and participants have one-to-one relationship. For a review of methods proposed for preventing other types of collusion, see [7].

INTER-PLAYER COLLUSION

If the content of the collusion agreement is an in-game resource, it is possible to detect by analysing the game session logs [2]. Detecting shared knowledge, however, is more difficult, because we can only observe the decisions made by the players in order to discern the intention behind the decision-making. To analyse this kind of collusion, we present a simple game, *Pakuhaku*, in the next section, but before that we need to consider two attributes of a game.

The first attribute divides games into *perfect information games* (such as chess), where the players can always access the whole game situation, and *hidden information games* (such as poker), where the players can access only a part of the game situation [6, §4]. Naturally, hidden information is worth colluding, because it gives the colluders benefit over the other players. But collusion is beneficial even in a perfect information game, because the decision-making process can always be seen as "hidden" information.

The second attribute is based on the properties of the game world, which can be *continuous* or *discrete*. If the central attributes of the game world are continuous, there usually is a well-defined metric to compute the distance between two game world locations. Since players try to dominate some geometric sub-area of the game world, the winnings of the game are related to the scope of the dominated area. Collusion can allow the players to govern a larger area than they would obtain by individual effort alone. If the game world consists of a set of discrete locations, the colluders can try to increase their joint probability of winning in the game by maximizing the subset of states they dominate.

When we consider the measuring and estimating collusion in some game environment, we could start by collecting real-world data for the purposes of analysis. However, it would be hard to ascertain what has been the driving force behind the human players at a given time. Another approach is to use synthetic players [5] some of which have been programmed to collaborate. Clearly, it is easier to create a large amount of test data with known co-operative properties with the latter approach, and we believe that it is the more fruitful one in this early phase of this research. The results obtained for artificial data should naturally be later evaluated and verified using real examples.

The idea behind our approach is:

(i) Generate game data with different number of players, colluders, game types, and collusion strategies.

(ii) Devise detection methods.

(iii) Run the detection method against the data to get results.

(iv) Compare accuracy: How many (if any) of the colluders got detected.

(v) Compare swiftness: How quickly the colluders were detected.

Naturally, this creates a competition surroundings where creating colluding synthetic players fights against devising detection methods.

In this paper, we limit ourselves to step (i). The subsequent steps will naturally be the focus of the our future work. Moreover, we intend to provide ready-to-use test data (akin to the Calgary Corpus [9]) for anyone interested in developing and testing their collusion detection methods as well as the possibility to fine-tune the synthetic players and develop new game types using our testbench system, *Pakuhaku*, which we will describe next.

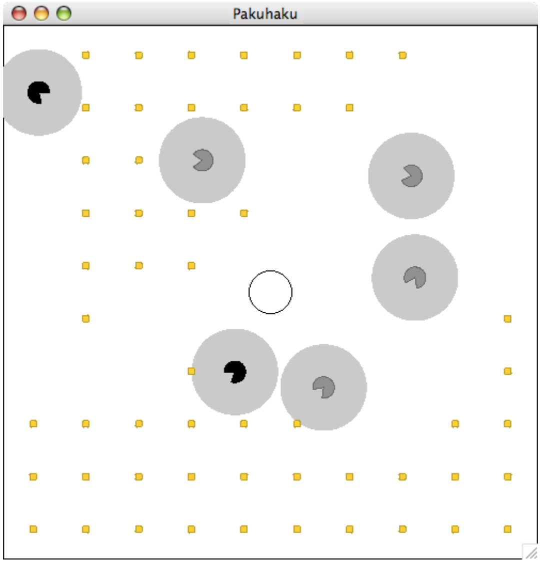

Figure 4: Screenshot from Pakuhaku with fog-of-war. The black players are colluding by dividing the game world into non-overlapping interest domains, while grey players search pills individually.

TESTBENCH FOR COLLUSION DETECTION

The basis for our testbench, *Pakuhaku* (see Figure 4), is the classical computer game *Pac-Man* [4]. We have omitted many features of the original game – such as the maze, ghosts, and power-up cherries – but we allow multiple players to take part in the game. Moreover, we have parameterized the number of directions the players can take (which ranges from three to infinite) and the area visible to the players can be limited by a fog-of-war. The goal of the game is simple: eat as many pills scattered in the game world as possible.

At each turn, each player makes a decision on the direction where to go. This decision is based on knowledge about the game world, which can be perfect (i.e., not limited by the fog-of-war) or hidden (i.e. limited to immediate surroundings by the fog-of-war). The system provides a communication channel, where the colluders can exchange one message in each turn. The communication can be used to assist, restrict and guide the co-colluders.

The game type can be one of the following:

- *Preset game world*: A given number of pills are positioned in the game world. The game ends when all the pills have been eaten.
  - Evenly distributed pills: The pills are positioned into rows and columns.
  - Randomly distributed pills: This pills are positioned randomly from a given distribution.
- *Regenerating game world*: Pills are repositioned to game world after they have been eaten. The game ends when the leader has eaten a given number of pills.
  - Dispenser competition: The game world has only one pill, which is dropped into a random position whenever it gets eaten.
  - Triple competition: A variation of the dispenser competition, where three pills are dropped randomly in a line somewhere in the game world.

Table 1: Results from 1,000 games with 100 randomly distributed pills and a fog-of-war. Of the five players *A* and *B* collude whereas *C*, *D* and *E* play fair. (a) The colluders share only knowledge of their whereabouts. (b) The colluders divide the playfield into interest domains. (c) Player *A* plays normally whereas player *B* tries to hamper the other players by following and eating pills in front of them.

test	player	min	max	mean	variance
	A	6	41	20.38	34.88
	B	5	44	20.14	35.85
(a)	*C*	5	42	19.91	36.77
	D	3	41	19.87	33.74
	E	5	41	19.69	34.34
	A	3	50	21.89	44.31
	B	5	47	20.20	39.36
(b)	*C*	3	40	19.40	34.21
	D	6	38	19.43	32.92
	E	4	40	19.07	35.42
	A	4	48	24.17	43.76
	B	2	37	13.65	27.41
(c)	*C*	3	47	21.19	56.95
	D	1	46	20.76	62.21
	E	1	55	20.23	58.99

The *Pakuhaku* system runs on the Java platform. The testruns can be done in a batch mode, where the system creates log data for further analysis (see Table 1 for a simple statistical analysis of the log), or the actions can be observed on screen. The player logic (including colluders and non-colluders) is freely programmable. The system can also be extended to include new game types.

DISCUSSION

Let us first consider collusion and the effect of the fog-of-war. If we have a perfect information game (i.e., no fog-of-war), colluders do not get any benefit by informing about the position of the game entities. Instead, they can agree on dividing the game world into non-overlapping interest domains (e.g., as a Voronoi diagram) so that each colluder eats the pills within the respective interest domain (see Figure 4). While non-colluders potentially target all available pills, thus competing with other non-colluders as well as with colluders, the colluders focus only to the subset that belongs to them and avoid competition with other colluders.

If the game has hidden information (i.e., the fog-of-war limits the visible area), the colluders get advantage by sharing the positions of the entities visible to them. This benefit is not as great in preset game worlds as in the changing ones. For example, in the triple competition, if the colluders know the position of two pills, the possible locations of the third pill are limited to a single line.

To present an example how collusion can be detected, let us consider the dispenser competition without the fog-of-war: Let the game area size be *A* and the number of players

p. Normal players would most likely try to balance between the following strategies:

- Patrol in the middle of the game field to minimize the average direction to a random location.
- Try to find an area that can be dominated and which is larger than A/p (i.e., an area where there are not many other players around).

In either case, whenever a pill drops, the player starts rushing to it. Note that if all players follow the same strategy, they have equal chances of winning ($1/p$). If both strategies are followed by some players, the latter strategy is the more profitable one.

Let there be q collusion participants, who divide the game field into q disjoint interest domains. If all non-colluders are patrolling in the middle, the colluders quite likely get most if not all the pills. Even in the latter case, the even distribution of colluders makes all areas of the game field equally uninviting to normal players, so their decisions will be more or less random.

CONCLUDING REMARKS

Collusion cannot be prevented, but some of its forms can be detected and punished afterwards. Therefore, the countermeasures are effective only if we can detect collusion accurately and swiftly. In this article, we focused on inter-player collusion and presented a simple game where collusion detection methods can be tested. The testbench creates game data that can be used to evaluate collusion detection methods. This paves the way to the future work, which will focus on designing detection methods, analysing them formally and improving them experimentally.

REFERENCES

[1] A. Ercole, K. D. Whittlestone, D. G. Melvin, and J. Rashbass. Collusion detection in multiple choice examinations. *Medical Education*, 36(2):166–172, 2002.

[2] U. Johansson, C. Sönströd, and R. König. Cheating by sharing information—the doom of online poker? In L. W. Sing, W. H. Man, and W. Wai, editors, *Proceedings of the 2nd International Conference on Application and Development of Computer Games*, pages 16–22, Hong Kong SAR, China, Jan. 2003.

[3] S. J. Murdoch and P. Zieliński. Covert channels for collusion in online computer games. In J. Fridrich, editor, *Information Hiding: 6th International Workshop*, volume 3200 of *Lecture Notes in Computer Science*, pages 355–369, Toronto, Canada, May 2004. Springer-Verlag.

[4] Namco. *Pac-Man*. Namco, 1979.

[5] J. Smed and H. Hakonen. Synthetic players: A quest for artificial intelligence in computer games. *Human IT*, 7(3):57–77, 2005.

[6] J. Smed and H. Hakonen. *Algorithms and Networking for Computer Games*. John Wiley & Sons, Chichester, UK, 2006.

[7] J. Smed, T. Knuutila, and H. Hakonen. Can we prevent collusion in multiplayer online games? In T. Honkela, T. Raiko, J. Kortela, and H. Valpola, editors, *Proceedings of the Ninth Scandinavian Conference on Artificial Intelligence (SCAI 2006)*, pages 168–175, Espoo, Finland, Oct. 2006.

[8] C. Vallvè-Guionnet. Finding colluders in card games. In H. Selvaraj and P. K. Srimani, editors, *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05)*, volume II, pages 774–775, Las Vegas, NV, USA, Apr. 2005.

[9] I. H. Witten and T. C. Bell. Calgary text compression corpus, accessed Aug. 16, 2007. Available at ⟨ftp://ftp.cpsc.ucalgary.ca/pub/projects/text.compression.corpus/⟩.

[10] R. V. Yampolskiy. Online poker security: Problems and solutions. In P. Fishwick and B. Lok, editors, *GAME-ON-NA 2007: 3rd International North American Conference on Intelligent Games and Simulation*, Gainesville, FL, USA, Sept. 2007.

[11] J. Yan. Security design in online games. In *Proceedings of the 19th Annual Computer Security Applications Conference (ACSAC'03)*, pages 286–297, Las Vegas, NV, USA, Dec. 2003.