Tregel, T., Göbel, S., Steinmetz, R. (2018, October). Role-based Multiplayer Content Online Adaptation in Large-scale Scenarios. In Proceedings of the 12th European Conference on Games Based Learning. Academic Conferences and Publishing International Limited.

Role-based Multiplayer Content Online Adaptation in Large-scale Scenarios

Thomas Tregel, Stefan Göbel, Ralf Steinmetz
TU Darmstadt, Darmstadt, Germany
thomas.tregel@kom.tu-darmstadt.de
stefan.goebel@kom.tu-darmstadt.de
ralf.steinmetz@kom.tu-darmstadt.de

Abstract: Many multiplayer games offer their players different distinct roles to choose from. Especially cooperative games use roles, as these foster cooperation between the players. Due to the complexity of possible group compositions their choice however, is often limited, as specific key roles are required based on the presented challenge. This limitation can severely reduce player enjoyment if it prevents them from playing their favourite role. Manually adapting every challenge for every possible group composition is infeasible due to the high amount of group compositions and their role interactions. These aspects are particularly relevant in the context of game-based learning, when adapting content to the current group of players.

To offer a feasible solution to the problem, this paper examines how player roles and interactions can be formalized and used in an automatic adaptation algorithm. Here, role properties and their interaction capabilities are used to achieve a balance against the game's tasks, which are adaptable parts of its content. This balance is modelled as an optimization problem, which allows for fast determination of a suitable content adaptation. The optimization problem can be solved using different approaches ranging from mathematical optimization to evolutionary algorithms depending on the given problem complexity.

In order to go from a theoretical model to an applicable algorithm a set of distinct properties is chosen that suffices to model some of the most complex mechanics present in role-based multiplayer games showing the versatility of the developed adaptation model. This provides game designers using the adaptation model a toolbox for flexible content creation.

This paper presents an adaptation system containing a mathematical formalization of player roles and game content as well as template modelling properties to assist in content creation. The system has been implemented in Matlab, connected to Unity and evaluated regarding different application scenarios using real data from current role-based games.

Keywords / Key Phrases: content adaptation, multiplayer, group modelling, mathematical optimization

1. Introduction

Many games offer their players many different roles to choose from. Especially cooperative games use roles, as these foster cooperation between the players. Massively Multiplayer Online Role-Playing Games (MMORPGs) often rely on complex interactions between different roles to create challenging content for the players. The most common roles in these games are tied to the so called *Holy Trinity* that refers to the three archetypes Tanks, Damage Dealers and Healers. Different examples could be found in sports or in collaborative learning environments, where each person has a distinct role. Roles within these games are identified and distinguished by the tasks they perform well and badly at.

The common aspect of these games is that players take on roles with different responsibilities and the game offers them challenges that they have to overcome by fulfilling their responsibilities. These challenges are often designed for specific group set-ups. Groups with a different distribution of roles have little to no chance of overcoming these challenges in the majority of cases. However, requiring specific group compositions to play a game successfully can be a problem, as it might force players to choose a role they do not want to play but that is needed to win the game, which might reduce player enjoyment.

Manually catering each challenge to every possible group would be infeasible. Therefore an adaptation model for cooperative role-based games was designed in Tregel et al. (2017) that models groups of players with respect to their roles and the interactions between the different roles. This allows game designers to model player groups accordingly, in order to adapt the respective game content, which can be both learning-focused and entertainment-focused.

2. Related Work and Problem Analysis

This section depicts a short summary of the given problem and the related work presented in our previous work in Tregel et al (2017).

Adaptation in games typically is used for changing specific aspects of a game in order to achieve a predetermined value on a given metric. The changes commonly affect aspects of a game that involve game mechanics and how the game is played more than visual aspects of the game.

Especially in singleplayer games adaptation is often used to adapt the game's content to the current user as seen for the Super Mario series (Pedersen et al. 2009, 2010, Togelius et al, 2011). As proposed by Yannakakis and Togelius (2011) for procedural content generation different types of evaluation functions can be used to incorporate the user's current status into the ongoing game. Many of these approaches used for content adaptation or procedural content generation are optimization problems, where the goal is to find the minimum or maximum for a specific measure, the objective function, for the given problem (J. W. Chinneck, 2006).

For multiplayer games adaptation is rarely used, both due to its complexity and due to the adaptation effects for other players. In a competitive scenario a personalized adaptation framework would handicap the dominant team in order to reach a balanced competition. For the sake of a balanced scenario this approach is applicable. As soon as monetary rewards are introduced the acceptance for a handicap diminishes. In terms of multiplayer player matching Konert et al (2013) has shown a suitable approach for multiplayer settings by matching the current player base to the given content. However, the content is limited to a group learning scenario, where people are matched together in a cooperative rather than a competitive manner.

3. Role-based Group Model

As a use case to test the developed adaptation model, a boss fight from the game World of Warcraft is modelled. World of Warcraft is a very popular Massively Multiplayer Online Role-Playing Game (MMORPG). As MMORPGs share several general gameplay mechanics, this section will start with modelling role and group properties and corresponding tasks that can be found in many of these games. Of course, every game has some specific properties and tasks that must be modelled individually, but these general properties and tasks can be used as a starting point for developers and then be expanded upon for their specific game. After establishing these general properties and tasks, the actual boss fight that will be adapted is described. Then it is shown how the different aspects of the fight pertain to the generic tasks and properties. Afterwards three specific tasks present in this fight are modelled. The section concludes with integrating the tasks and properties into the adaptation model. The overall optimization workflow for the model is depicted in Tregel et al (2017).

3.1 Role Properties

To model all of the classes and properties that can be found in many MMORPGs would go greatly beyond the scope of this paper. Instead, a set of distinct properties has been chosen that will suffice to model some of the most complex mechanics present in these games and thus show the versatility of the developed adaptation model. Abilities and attributes are not modelled individually but directly mapped on role properties for simplification.

- Effective Health: This property describes how much damage a player can survive. A higher value means that more damage can be survived. Health, armour and damage mitigation abilities of an avatar can be integrated into the effective health.
- Maximum Sustainable DPS Single Target: The maximum damage per second (DPS) this role can achieve and sustain indefinitely to a single target within range.
- Maximum Sustainable DPS Multi Target: The maximum damage per second this role can achieve and sustain indefinitely to a group of targets within range.
- Dispel Rate: The rate at which this character can dispel certain effects, i.e. remove the effect from a target. This property is represented as the number of dispels per second. It can be used to estimate how often enemies cast effects that can be dispelled without overwhelming the group.
- Interrupt Rate: The rate at which this character can prevent that an enemy from casting a spell. This property is represented as the number of interrupts per second.
- Tanking: This property indicates if this character can act as a Tank, i.e. ensure that the enemies attack him instead of his allies.

- Movement Impairment: Certain abilities are only usable when the player stands still. This property describes by what ratio the damage or healing per second of this character are reduced when he is forced to move around. A value of zero denotes that this role has no impairment while moving, while a value of one signifies that the role can do nothing while moving.
- Maximum Sustainable Heal Group: The maximum sustainable healing a character can apply to a group over a long period of time.
- Maximum Sustainable Heal Single Target: The maximum sustainable healing a character can apply to a single target over a long period of time.
- Additional Burst Heal HPS: The amount of health points (hp) that this character can apply instantly to a group of players with his burst heal ability. The amount of hp is restored to every player of the targeted group.
- Additional Burst Heal Charges: How often the character can apply his large burst heal.

The burst heal properties have been included to show that the adaptation can handle limited resources on the group's side. In the model for the boss fight, the movement impairment property will only be applied to healing, not to damage dealing. In this way, the ability to model nonlinear relationships between tasks and group properties can be shown without overcomplicating the model.

3.2 Group Properties

In the following two Sections we will present a set of group properties and tasks we associated with the presented role properties in Section 3.1. The given formulas are presented with text based variables in order to improve traceability for users, as our system is used in a learning and game environment for character performance modelling, which we present in Section 5. Notation-wise summations are listed as follows:

$$Property = \sum_{players:p} Attribute(p)$$

In this example the summation functions as an iterator over all available players, storing the current iteration variable *p*, which is used in the respective formula's attribute. The group properties computed from the role properties are as follows:

• Group HPS: HPS stands for healing per second. This is the amount of health the group can on average restore each second. This can be computed from the sustainable single target and group heals. For group heals the average number of players that will benefit from the respective heal must be considered. Also single target and group heals cannot be applied both to their full extent all the time. Thereby they are mutually exclusive.

$$GroupsHPS = \sum_{players:p} \begin{matrix} \alpha * MaximumSustainableHealSingleTarget(p) + \\ (1-\alpha) * averageNrOfTargets * MaximumSustainbleHealGroup(p) \end{matrix}$$

With α being the fight's ratio of SingleTarget and MultiTarget healing.

• Available Burst Heal: The average amount of health that can be restored to a group of players using burst heal abilities. The amount of health is restored to every player of the targeted group.

$$AvailableBurstHeal = \sum_{players:p \text{ with } AdditionalBurstHealCharges>0} AdditionalBurstHealHPS$$

• Maximum HPS: The maximum amount of health that can be restored to a group of players using basic heals and burst heal abilities. The amount of health is restored to every player of the targeted group.

$$MaximumHPS = GroupsHPS + AvailableBurstHeal$$

• Movement Impairment The average impairment of healing if movement is forced on the players.

$$MovementImpairment = \frac{1}{numberOfPlayers} \sum_{players:p} MovementImpairment(p)$$

• Minimum Effective Player Health: The lowest health present in the group. This can be used to ensure that damage from enemy attacks is not instantly lethal to the players. It is most useful for attacks that target a random player or groups of players.

$$MinimumEffectivePlayerHealth = \min_{players:p} EffectiveHealth(p)$$

• Effective Main Target Health: The health of the player that will be targeted by the enemies. This is assumed to be one of the Tanks or in the case that no Tank is present, the player with the highest health. Since Tanks normally can take more damage than an average player, this property can be used to set the attack damage for attacks directed upon the main target of enemies.

$$EffectiveMainTargetHealth = max(\max_{players:p\ with\ tanking=1} EffectiveHealth(p), MinimumEffectivePlayerHealth)$$

• Dispel Rate: The rate at which effects can be dispelled.

$$DispelRate = \sum_{players:p} DispelRate(p)$$

• Interrupt Rate: The rate at which casts can be stopped.

$$InterruptRate = \sum_{players:p} InterruptRate(p)$$

• Maximum Sustainable DPS: The maximum amount of damage per second the group can sustain over a longer period. For attacks that target multiple enemies, the average number of targets hit must be considered. As for the healing abilities, a split between single target and multi target abilities is assumed. Additionally it is assumed that if players have melee and ranged abilities that they will use the stronger ones and ignore the other abilities.

$$MaximumSustainbleDPS = \sum_{players:p} \begin{matrix} \beta * MaximumSustainableDPSSingleTarget(p) + \\ (1-\beta) * averageNrOfTargets * MaximumSustainbleHealMultiTarget(p) \end{matrix}$$

With β being the fight's ratio of SingleTarget and MultiTarget damage requirements.

3.3 Tasks

In this section tasks that can be found in many MMORPGs are described with an example of how Enemy Damage is integrated.

The *Enemy Damage Per Second* (*Enemy DPS* in short) task accumulates all the damage players will receive. The damage is considered on a per second view, as it can be easily related to the amount of healing the group can do per second. In this way, it can be modelled if the group can sustain the damage dealt throughout the fight. The *Enemy DPS* can be calculated as the sum over the DPS of all damage sources present in a level:

$$EnemyDPS = \sum_{DamageSources:ds} DPS(ds)$$

These damage sources will in most cases be enemy abilities. Every ability may cause damage directly and also have an additional damage effect. Abilities normally have cooldown, which determines how much time must pass before it can be activated again. An ability's DPS can be calculated as:

$$DPS = (directDamage + additionalDamage)/cooldown$$

The *Required Movement Rate* determines how often players will be forced to move, e.g. to avoid an attack. The actual units used for this rate may differ from game to game, but a per second rate will be

reasonable for most computational purposes. The overall Required Movement can be calculated in the same way as the *Enemy DPS*:

$$RequiredMovement = \sum_{Sources:s} RequiredMovementRate(s)$$

Again, the sources will in most cases be enemy abilities. For the movement rate only the cooldown of the ability and how often one use of the ability will force players to move is relevant. An ability's Required Movement Rate can be calculated as:

$$RequiredMovementRate = numberOfForcedMovements/cooldown$$

Enemy DPS and *Required Movement Rate* are opposed to the group property *Group HPS*. This is the case since all the damage done must be healed in order to survive the fight and the enforced movement reduces the healing of some roles. The more often players are forced to move around, the stronger the effect of their movement impairment is. The balance is calculated as:

$$EnemyDPS - GroupsHPS * (1 - RequiredMovement * MovementImpairment)$$

Note that if either no player has a movement impairment or the players are not forced to move around, there is a one-to-one relationship between the damage done by the enemies and the healing the group can provide.

A Tank is needed to focus the enemies' attacks on one person who can withstand them. However, there are situations were multiple Tanks are needed. Considering additional enemies (called Adds), one has to keep in mind that a Tank may be to able handle several smaller enemies but only one big enemy. Therefore each enemy can be associated with a value denoting how much of a Tank's attention it requires. The extent of this task can be calculated as:

$$Enemies = \sum_{enemy:e} attentionFactor(e)$$

To dispel an effect, i.e. remove it from the target, a player needs to have an ability that allows him to do that and the ability must be usable when the effect needs to be dispelled. Since abilities have a cooldown, i.e. they need some time after being used before they can be used again, the rate with which effects need to be dispelled should match the rate with which the players can dispel them. To calculate the rate of dispellable effects, the cast rates of all abilities that cause effects that can be dispelled can be accumulated:

$$RateOfDispellableEffects = \sum_{DispelSources:ds} CastRate(ds)$$

The *Rate of Interruptible Casts* can be calculated analogue for interruptible casts.

Burst Damage is damage that can be inflicted instantly to a target. This task models the maximum damage that can be instantly inflicted upon a group of player. This is modelled, as group damage can be healed with Area-of-Effect healing abilities. These abilities often need a long time to recharge (sometimes several minutes), which makes them a valuable and limited resource during a fight.

$$TotalBurstDamage = \sum_{DamageSources:ds} CastRate(ds) * TimePeriod$$

With the average burst damage being computed as:

$$AverageBurstDamage = (TotalBurstDamage)/TimePeriod$$

Maximum(Non)TankDamage is a task that measures the maximum damage the respective player can receive from a single attack. This is important since if this was higher than the health of a player, it would simply kill the player instantly, making the fight impossible. Tanks and other players are handled differently but have analogue tasks, since Tanks normally have a lot more health than other players.

This task could be calculated as the maximum over all possible damage sources that target a random player or specifically a player that is not a Tank. This however, would result in a non-smooth function, as the maximum function is not smooth. A better way to handle this task is to model it as a collection of constraints. For each source of damage one constraint comparing it to the Minimum Health is used. In mathematical terms, our model will be subject to:

$$\forall_{DamageSources:ds} MinimumHealth \geq damage(ds)$$

4. Evaluation

World of Warcraft is one of the most popular MMORPGs. It has an enormous complexity and features some of the most intricate mechanics for cooperative games. As a use case, a boss fight from this game is modelled and the adaptation model is used to provide possible adaptations for different group compositions. This part of the evaluation is of theoretic nature since the game does not offer any adaptation for comparison and does not offer any interfaces to connect the adaptation to the game. This example has been chosen to show that the algorithm can be used even in very complex games, as World of Warcraft is one of the most complex games featuring roles in a cooperative scenario and this boss fight uses many of the most complex mechanics present in the game.

An in-depth description of the encounter can be found on www.icy-veins.com, for example. In the chosen fight, a group of players fights Tyrant Velhari, a boss enemy in Hellfire Citadel. Tyrant Velhari uses three abilities throughout the entire fight, along with her standard melee attack. These abilities are:

4.1 Role and Group Property data

For this use case the three main roles present in World of Warcraft are considered: Tank, Healer and Damage Dealer. As there is an abundance of variations to these roles available and not all of these can be modelled in the scope of this paper, nine specific roles have been modelled using the properties established in Section 3:

- Mitigation Tank based on Protection Warrior.
- Restoration Tank based on Blood Death Knight.
- Single Target Melee DPS based on Subtlety Rogue.
- Multi Target Melee DPS based on Combat Rogue.
- Single Target Ranged DPS based on Arcane Mage.
- Multi Target Ranged DPS based on Fire Mage.
- Single Target Healer based on Holy Paladin.
- Multi Target Healer based on Mistweaver Monk.
- Healing over Time Healer based on Restoration Druid.

The values for healing and damage were extracted from the website WarcraftLogs[1] for boss fights in Hellfire Citadel. These values are an image of the whole content's lifespan, as they were extracted after the release of new content.

4.2 Balance Functions and Constraints

These are the six balance functions derived from the tasks and group properties described above. The aura is the specific aura for each phase. The other functions are always accumulated over the abilities active in the given phase, as illustrated in the tasks section.

$$balance(parameters, group) = \begin{pmatrix} EnemyDPS - GroupHPS * (1 - RequiredMovementRate * MovementImpairment \\ RateOfDispellableEffects - DispelRate \\ RateOfInterruptableCasts - InterruptRate \\ Enemies - Tanking \\ TotalBurstDamage - TotalBurstHeal + AverageBurstDamage - AverageBurstHeal \\ EncounterIntensity - EncounterCounterAbility \end{pmatrix}$$

The constraints are on the one hand the tasks that are modelled as constraints and on the other hand the bounds that are enforced on the tasks. For the tasks modelled as constraints, the respective

1 https://www.warcraftlogs.com/statistics/8#aggregate=amount&boss=1785&metric=bossdps&class=Any

terms are used, and each of the constraints is split into as many single constraints as there are terms to render the maximum function unnecessary. The resulting constraints are:

$$constraints(parameters, group) = \begin{pmatrix} MaxmimumNonTankDamage - MinimumHealth \\ MaximumTankDamage - TankHealth \\ TotalBurstDamage - AvailableBurstHeal \end{pmatrix}$$

4.3 Use Case Evaluation

The developed optimization problem features 14 group properties, all six balance functions shown in Section 4.2, their respective constraints and contains 41 parameters. Comparing this optimization problem to the test cases used for the evaluation of the algorithms in Tregel et al. (2017), it can be seen that the example does not have clear similarities to one specific test set. This makes it more difficult to form an expectation of how the different algorithms might perform. As context for the use case, an offline adaptation during loading is assumed and constraints should be satisfied, as a violation for some of them would render the fight unwinnable.

The requirement to satisfy the constraints excludes simulated annealing. Since the adaptation is done during loading, the higher computational time required by the genetic algorithm over Fmincon is neglectable. Therefore, the genetic algorithm with Patternsearch as hybrid function is chosen for the evaluation of the use case.

4.4 Role Distribution

Concerning different role distributions the quality of the solutions is examined as well as specific parameters that showed interesting behaviours across the different groups.

For all examined examples, two different plots are presented. The first one is a three dimensional plot representing the values as height. The second one represents the same values as a grid, where each cell represents one group. Both graphs use the same colour scheme to represent values. The colours go from dark blue, representing the lowest values, to a dark red that represents the highest values. For all graphs only the lower left triangle represents the actual group combinations. The upper right triangle shaded in a dark blue is an artefact of the fact that a rectangular matrix is expected by the plotting function.

Along the horizontal axis (showing to the right for the three dimensional plot) the amount of Healers in the group is presented. Along the vertical axis (showing to the left for the three dimensional plot) the amount of Tanks. The amount of Damage Dealers in the group is implicitly given, as there are always as many Damage Dealers as are players needed to reach the group size of twenty five people after adding the Tanks and Healers.

Note that the leftmost column does not represent actual results, as for the groups in this column no feasible solution was found. This is caused by the constraints on Burst Heal Charges. In the specified column are all groups that have no Healers. Since only the Healers were modelled with Burst Heals, these groups can not satisfy the constraints. This shows that the encounter does not cater to all possible group compositions. If it is desired that any kind of group has a chance to win this fight, either the mechanics must be changed or the constraints regarding Burst Heals must be remodelled to consider players without explicit Burst Heals.

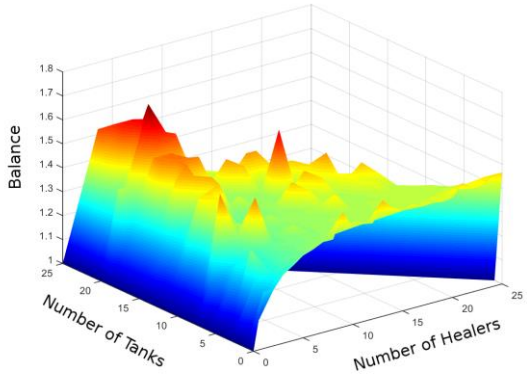

Figure 1: Heightmap of the solution's quality.

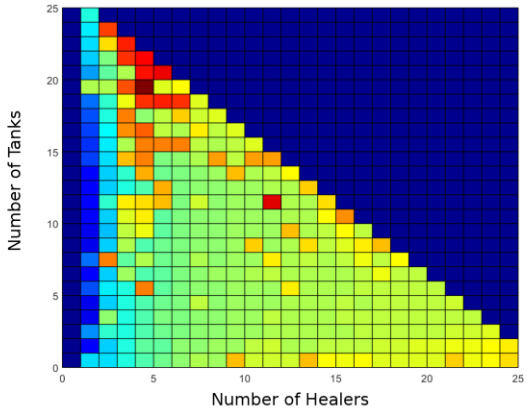

Figure 2: Grid representation of the solution's quality.

The quality of the solutions found is a good measure to evaluate if different role distributions will have equal chances of winning the fight. Figure 1 shows the value of the target function at the found solution as a three dimensional height plot. Figure 2 represents the same values purely as colours. Higher values in this case represent less balanced scenarios, i.e. too difficult or too easy. In general, the plot shows quite similar results for most groups. Only in the region where the groups have many Tanks and only a few Healers and Damage Dealers are several solutions that have a significantly worse balance than the other scenarios. A designer attempting to achieve an overall equal balance could use these results to examine specifically these solutions and adapt the mechanics accordingly.

Upon examination, the parameters show quite different behaviour from each other. Many parameters show no correlation in the found solutions regarding the group composition. The main reason for this is that there will be several different minima of the target function with different parameter set-ups leading to similar results. For example, increasing the damage of an attack has the same result as reducing the attack interval. Both increase the overall damage inflicted on the players. The use of the genetic algorithm as solver highlights this property of the model, as its random nature can drive it to different solutions.

There are some parameters that actually show a consistent behaviour despite the randomness involved in finding the solution. These behaviours highlight certain properties of the model and the parameters. Some examples of these correlations will be shown for selected parameters.

The first one is Tyrant Velhari's health. Figures 3 and 4 show the values for the different group compositions. It can be seen that with a larger amount of Healers and Damage Dealers the value for the health rises. This will most likely be the case as these group compositions can deal more damage and heal more damage received. This can be compensated by extending the length of the fight, which is mostly dependant on Tyrant Velhari's health.

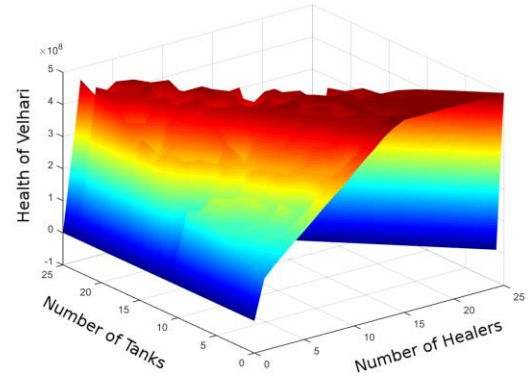

Figure 3: Heightmap of Tyrant Velhari's health.

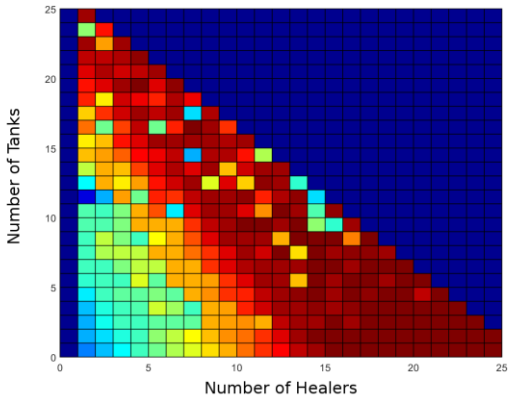

Figure 4: Grid representation of Tyrant Velhari's health.

Another example is the amount of *Ancient Enforcers*, the additional enemies that are spawned during the fight. Figures 5 and 6 show the amount of *Ancient Enforcers* spawned for the different group compositions. It can be seen that the amount of Enforcers spawned increases linearly with the number of Tanks present in the group. More Tanks can take on more Enforcers the adaptation can only try to balance this by spawning more Ancient Enforcers.

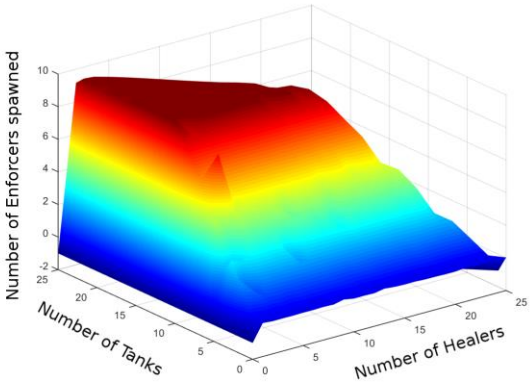

Figure 5: Heightmap of the number of additional enemies spawned.

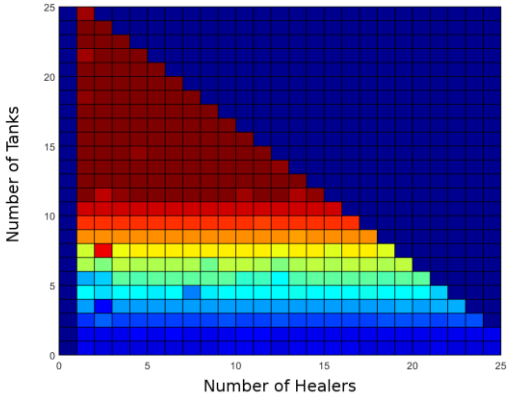

Figure 6: Grid Representation of the number of additional enemies spawned.

The evaluation of the use case has shown that the adaptation model can be applied to a realistic scenario. The usability of the model proved to be good, as a very complex scenario could be modelled and the results overall are promising. The overall balance achieved was very similar for different group compositions and different role versions, indicating a well-balanced system. For small group sizes and only a few roles an extensive analysis of the different parameters can be done. The evaluation with twenty five players took about ten hours to run and delivered various insights regarding parameter bounds and how the group composition affected certain parameters. Such an analysis might also reveal group compositions that are not well balanced and need further consideration.

However, any algorithm using randomness can deliver very different solutions during different runs. Should this pose a problem for a specific game, one possibility would be to employ Fmincon as solver. Since it does not use randomness, it will not deliver different solutions for the same problem in different runs. If however local minima are close to each other, it might still deliver quite different solutions with only little change in the group composition.

5. Unity Integration

The whole system has been integrated into the Unity game engine into a cooperative role-based game shown in Figure 7. This game contains the basic interactions of classic RPGs, combined with a character editor in order to create and customise the own character in regards to attributes, abilities and their respective improvements.

The game, which we will not go into detail to, is designed as a couch coop game, allowing up to eight players (due to UI layout limitations) to play on the same screen using standard input devices. After the character creation our adaptation system is initiated in order to calculate a fitting adaptation to the chosen group. Therefor the available bosses and challenges have been modelled according to the previously described properties and tasks. As shown in our previous work (Tregel et al., 2017) the required adaptation time for the given problem size lies in the segment of low double digit number of seconds and is thereby applicable to be used during a short loading screen.

In the context of a learning environment the system allows for easy modelling of individual character roles or classes. Users can apply, create and modify character and group properties to selected tasks in order to automatically change the game's balancing and thereby its difficulty.

Figure 7: Unity game using our adaptation system in a three player scenario.

6. Conclusion

In this paper an adaptation system containing a mathematical formalization of player roles and game content as well as template modelling properties to assist in content creation was developed. Based on this foundation the system has been implemented in MATLAB and connected to Unity

After establishing the properties and restrictions of the optimization problem the system was implemented in MATLAB. It contains a list of basic properties and tasks usable for all types of cooperative role-based games. In order to evaluate both the system's performance and functionality a complex application scenario from a *World of Warcraft* encounter has been modelled, implemented and evaluated. The evaluation showed a stable performance for typical group compositions encountered within these games. For more exotic compositions like trying to play only one healer in a 25 player scenario the system still delivered good results, while exhausting the model's adaptation boundaries.

References

Chinneck, J.W., 2006. Practical Optimization: a Gentle Introduction. *Systems and Computer Engineering*, Carleton University, Ottawa.

Konert, J., Burlak, D., Göbel, S. & Steinmetz, R., 2013. GroupAL: ein Algorithmus zur Formation und Qualitätsbewertung von Lerngruppen in E-Learning-Szenarien mittels n-dimensionaler Gütekriterien. *DeLFI Vol 13*.

Pedersen, C., Togelius, J. & Yannakakis, G.N., 2009. Modeling Player Experience in Super Mario Bros. *IEEE Symposium on Computational Intelligence and Games*. CIG 2009. pp 132–139.

Pedersen, C., Togelius, J. & Yannakakis, G.N., 2010. Modeling Player Experience for Content Creation. *IEEE Transactions on Computational Intelligence and AI in Games* 2(1), pp 54–67.

Togelius, J., Kastbjerg, E., Schedl, D. & Yannakakis, G.N., 2011. What is Procedural Content Generation? Mario on the borderline. *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games*. ACM.

Tregel, T., Alef, J., Göbel, S. & Steinmetz, R., 2017. Towards Multiplayer Content Online Adaptation using Player Roles and their Interactions. European Conference on Games Based Learning. Academic Conferences International Limited.

Yannakakis, G.N. & Togelius, J., 2011. Experience-Driven Procedural Content Generation. *IEEE Transactions on Affective Computing*, 2(3), pp 147–161.