Bringing New HOPE to Networked Games: Using Optimistic Execution to Improve Quality of Service

Ryan Hanna, Michael Katchabaw
Department of Computer Science
The University of Western Ontario
London, Ontario, Canada
Tel: +1 519-661-4059
rhanna@uwo.ca, katchab@csd.uwo.ca

ABSTRACT

As more games of a wider variety of genres move online to provide multiplayer experiences to their players, there is an increasing need to improve the quality of service delivered to the players of these games. Players tend to have the same performance and consistency expectations of their online multiplayer games as they do of their single player games, without realizing the issues and problems introduced by networking their games together. This results in a tremendous challenge for developers of networked games, because issues such as latency work strongly against meeting the needs of players.

In this paper, we discuss the concept of optimistic execution to help game developers mask or hide the effects of latency in their networked games. We introduce the notion of optimistic execution, present our work in this area, dubbed New HOPE, and comment on its ability to assist game developers in this important area.

Keywords

Latency, lag, and delay reduction, optimistic execution, quality of service.

Proceedings of DiGRA 2005 Conference: Changing Views – Worlds in Play.

INTRODUCTION

Despite improvements to networking technologies, adverse network conditions continue to have serious consequences for online games. With online games one of the fastest growing industry segments [7], and with players bringing the same quality of service expectations from their offline experiences to their online experiences, the challenges faced by developers of networked games are only going to increase.

Latency (also commonly referred to as lag or end-to-end delay) is an especially challenging problem [2], leading to anything from minor annoyance to a totally unplayable experience. In some respects, unfortunately, there is little that can be done, particularly for games played over wide area networks, such as the Internet. Ultimately, the speed of light is not amenable to change.

Action-oriented games are hardest hit by latency issues. Recognizing this, there have been several proposed solutions, many of which focussed at shooting aspects of first-person shooters, such as [1,5]. Unfortunately, these solutions tend to be very narrow and very ad hoc, applying only to one aspect of a single genre of games. They also tend to either induce confusing gameplay or introduce potential inconsistencies that can break immersion in the game quite easily [2]. Using client side prediction to hide movement-related latency is a common technique, but ultimately has limitations [4]. With more varied gameplay moving online, a more general, flexible, and robust solution is necessary.

Our current work introduces a new formalism to networked games: optimistic execution. The basic premise behind optimistic execution is to allow certain game activities to occur without checking with other parts of the game first, provided that the outcomes of the activities are predictable and recoverable, in case predictions turn out to be incorrect once synchronization occurs. Optimistic execution of such activities occurs in parallel with confirmation of their outcomes, allowing the latency of synchronization to be effectively hidden from the player.

Elements of optimism can be found in earlier work in this area, but in an ad hoc fashion, without the formalisms and support necessary to properly make use of it. By providing appropriate design methodologies and programming constructs, developers have complete control over how optimism is used within their games, and can even cede some of this control to the games, or their players. Consequently, optimistic execution has the potential to provide the latency solution desperately needed.

We begin this paper with a discussion of the original HOPE, which served as a starting point for our current work. We then present New HOPE, an evolution of HOPE specifically targeted towards the needs of networked games. We discuss a proof of concept of New HOPE, as well as work in progress towards providing tools support to game developers for optimistic game development. We finally conclude our paper with a summary, and discussion of future work in this important area.

HOPE

HOPE (Hopefully Optimistic Programming Environment) [3] was developed at the University of Western Ontario to bring optimistic execution to networked applications, before the current surge in popularity of networked multiplayer games. It was intended to support non real-time applications such as banking systems and other transaction-oriented systems in which a complete rollback of activity could be accomplished when an optimistic prediction was discovered to be incorrect.

Consider, for example, the traditional (pessimistic) execution scenario shown in Figure 1:

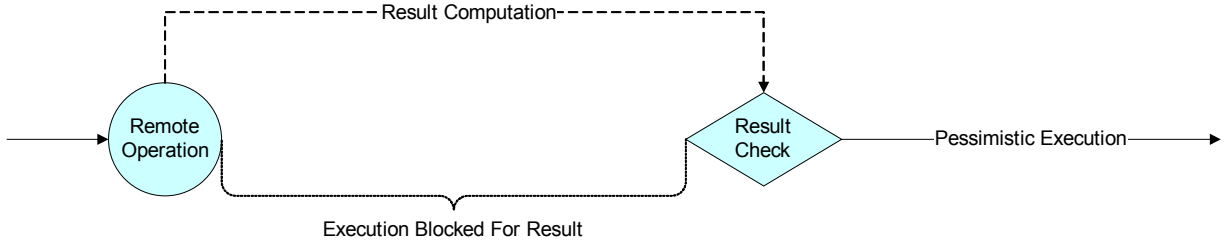

Figure 1: Traditional, pessimistic execution.

When program execution reaches a remote operation, it must block and wait for the results of the computation to be executed remotely before proceeding with execution. Execution in this case is considered pessimistic in that it must be certain of the results checked before proceeding. If latency is high, it can take considerable time to receive the necessary results; in the mean time, program execution has been suspended. For example, if a player was moving in a multiplayer game, and every movement required a synchronization check before it was permitted, the performance of the game would be seriously affected.

If the result generated can be predicted with reasonable certainty, optimistic execution could be used to mask the latency and allow computation to proceed while the result was still being calculated. To support this, HOPE introduced the concepts of guesses, affirmations, denials, and rollbacks, among other formalisms [3], as shown in Figure 2:

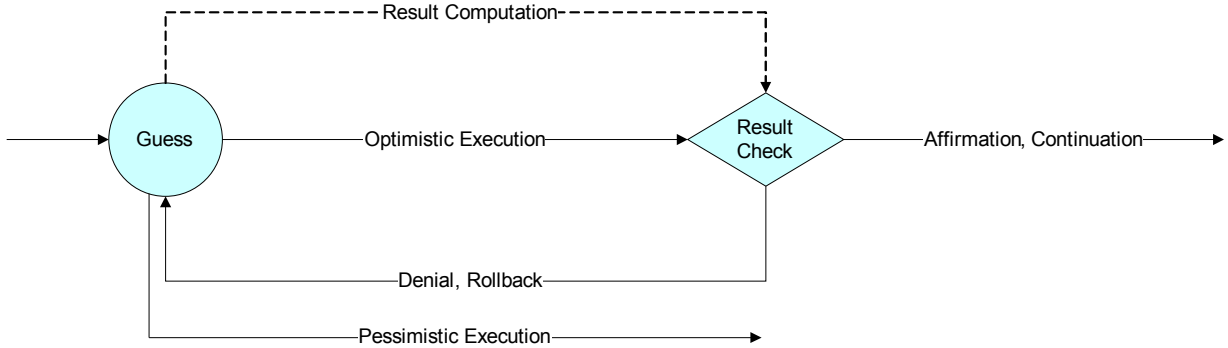

Figure 2: Optimistic execution in HOPE.

The remote operation in this case begins with a guess. If we presume that the value of the result is highly predictable, this value is guessed and assumed to be valid. Optimistic execution of the

program proceeds based on this assumption until the result is computed remotely and is made available to the program. When the actual computed result is checked against the assumed result, there are two possible outcomes. The first possibility is that there is a match, and the optimistic assumption has been affirmed. In this case, execution can proceed without alteration, and the latency of the computation and communication involved has been effectively hidden. The second possibility is that the actual result and assumed result do not match. In this case, the optimistic assumption is denied, and all execution that happened in the mean time is rolled back until the program is in the same state it was when the guess was originally made. At this point, the program has no choice but to continue execution as if the guess was never made, and the program was executing using the pessimistic model in Figure 1. In doing so, however, no time has been lost, except for the overhead involved in the rollback.

While suitable for transaction-oriented applications, HOPE is not suitable for networked games. These games are very much real time interactive multimedia applications, and the methods employed by HOPE for managing optimistic execution will not work well in this environment. Ultimately, a total rollback does not make sense for networked games. This would be tantamount to undoing player actions and reactions, moving the game backwards in time. This would break immersion and believability in the game, and is highly undesirable. For example, undoing the firing of a weapon in a game could be very problematic.

NEW HOPE

While HOPE cannot be directly applied to networked multiplayer games as is, it still contains many of the philosophies and theories required to support optimistic execution in a compatible fashion. New HOPE is a new embodiment of HOPE that is specifically designed for real-time applications and the needs of networked multiplayer games, as discussed in the sections below.

Programmable Optimistic Execution

If the results of activities are predictable and recoverable, then execution is allowed to proceed without first checking the results of the activities, much as in the original HOPE The results are checked when they become available, without stalling the game to wait.

Recovery, Not Rollback

Unlike HOPE, which requires a full rollback for every optimistically executed action, New HOPE requires recoveries. A recovery is a method of handling an incorrect optimistic prediction while keeping the flow of gameplay moving forward. When completed, a recovery procedure must bring the game into a consistent and acceptable state.

Consider the new optimistic execution scenario in Figure 3. The primary difference between this scenario and the scenario in Figure 2 is the use of recovery on the denial of a guess, instead of rollback. By carefully selecting a recovery appropriate to the game context, execution can proceed forward without having to go back first. (If multiple recovery methods are possible, one can be selected randomly, using a weighted process, or based on the operation and actual results received.) When a guess is incorrect, the optimistic execution depending on that guess is also incorrect, and the recovery attempts to correct this situation as best as possible. For example, if a player in a game was attempting to jump onto a moving object, it may be reasonable to assume that they made the jump. If it is determined the jump was missed after checking the result, a

suitable recovery could be to have the player slip and fall from the obstacle to place the player in a consistent state in the game.

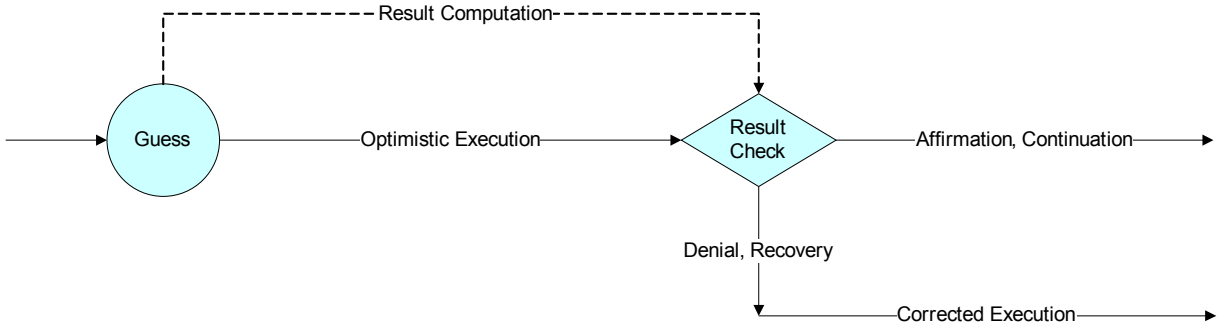

Figure 3: Optimistic execution in New HOPE.

While optimistic execution in this fashion may allow incorrect computations to occur while in a speculative state, it is better than the alternatives. Pessimistic execution forces the player to experience the latency involved, while optimistic execution hides it. A controlled recovery is better than a rollback, and is better than either no recovery or an ad hoc or uncontrolled recovery. If guesses are made when appropriate, and recoveries in place are suitable, then the optimistic execution of New HOPE will work quite well. If, on the other hand, outcomes of a remote operation cannot be predicted, or if the actions taken while in a speculative state cannot be easily recovered from, then optimistic execution could be hazardous and lead to undesirable effects.

Flexibility in Control of Optimistic Execution

To provide more flexibility in control over what executes optimistically in a game, new optimism primitives were added to New HOPE. One primitive, padding, is depicted in Figure 4.

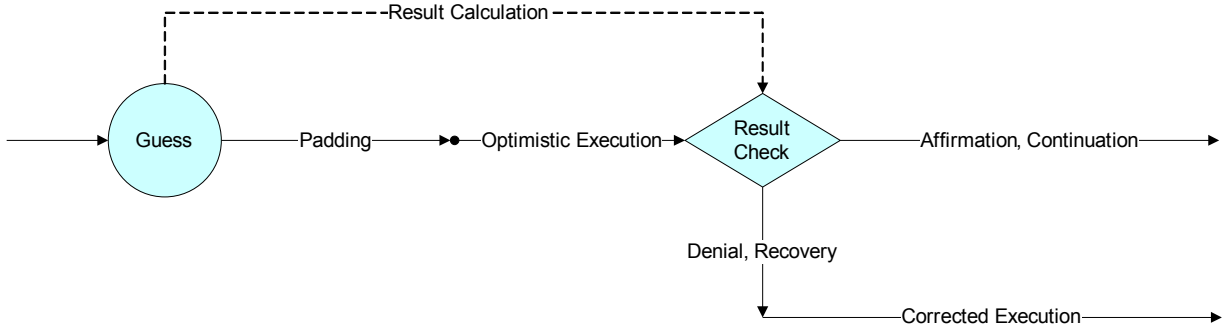

Figure 4: Optimistic execution in New HOPE with padding.

Padding is, in essence, a distraction element added to hide latency before proceeding with optimistic execution in cases where guesses are uncertain or recoveries are difficult. By using padding, the amount of optimistic execution can be limited to what is acceptable under the circumstances, while at the same time hiding the inherent latency in the remote operation. If the length of the padding element is equal to or greater than the latency of the operation in question, then an essentially pessimistic execution can begin after the result check, without the delays made apparent by a pessimistic execution. For example, if a player wanted to pick up an item and add it to their inventory, an animation could be played of the player's character bending over

to pick up the item, while the corresponding remote operation occurs in the background. In this case, the animation distracts the player from recognizing the latency involved in the operation.

Another primitive added to provide more control over optimistic execution comes in the form of synchronization points. A synchronization point can be added to either check on the status of a remote operation (non-blocking mode) or to force a wait for the remote operation to complete (blocking mode). This can be used to prevent further optimistic execution from proceeding if that execution would be difficult to recover from. It is important to note that recovery would still be necessary upon denial for any optimistic execution up until this point, however. This is shown in Figure 5.

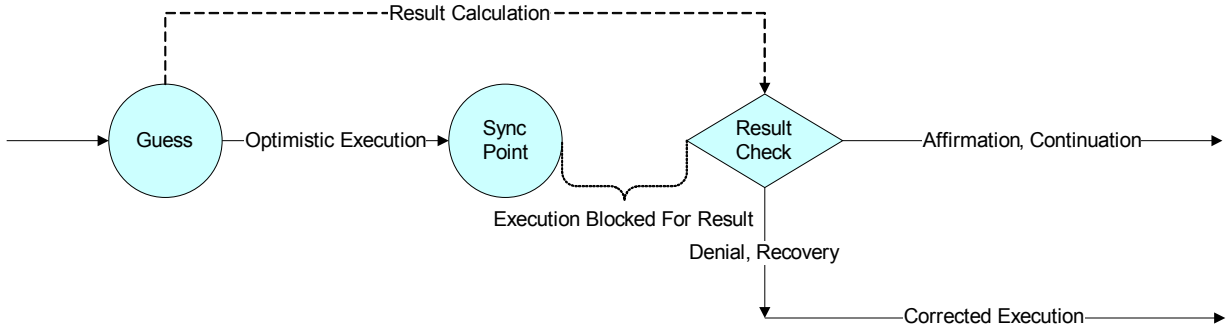

Figure 5: Optimistic execution in New HOPE with synchronization points.

Variable Predictability

The state of a game can be used to make optimism decisions dynamically, unlike HOPE. When game state makes a certain action more predictable, then that action can be completed optimistically. When game state makes the results of an action less certain, for example below an acceptable threshold, then the action should be carried out pessimistically, waiting for the results before continuing. For example, as the number of players in an area of the game world increases, the less predictable the situation becomes; the fewer the number of players, the more predictable outcomes are. Revisiting the previous example in picking up an item, the chances for success are extremely high if no one else is in the area. If there are several other players in the area, however, it becomes more possible that another player would attempt to pick up the item at the same time, and the end result becomes less predictable.

The predictability threshold mentioned above can be presented to users as a tuneable game parameter. Players that would ordinarily experience considerable latency within a game can set the threshold lower, allowing more operations to occur optimistically to reduce the effects of lag. (More recoveries would be required in such a case, however, as predictions made could be less certain than before.) Players encountering less lag can keep the threshold higher to avoid the need for recoveries. Ideally, the game can tune this threshold itself dynamically in response to observed network latency, but it is also good to give players control over their experiences.

EXPERIENCES WITH NEW HOPE

In this section, we briefly describe a proof of concept for New Hope, and present a discussion of our experiences to date.

Proof of Concept

As proof of concept, we implemented new optimistic algorithms in the Zen of Networked Physics simulation environment [4]. This environment provides the ability to simulate a cube moving around an environment with a physics model applied, along with a projection of the cube's state to a server and remote client. Varying degrees of latency and packet loss can also be applied to observe first hand the effects of network conditions on the experience viewed locally and remotely. This environment also provides algorithm components for client side prediction, smoothing (to help recover from the inevitable snapping in client side prediction), and forward error correction (to reduce the impact of packet loss). Consequently, this environment provides a solid experimental framework for investigating optimism in networked games.

Using elements from New HOPE, and the facilities from the simulation environment, we constructed several new optimistic algorithms. This includes algorithms for client side adjustment of player state, and a client projection of state. The former algorithm used padding to adjust the momentum of a cube to allow a portion of latency to be masked by a more realistic responsiveness of the cube, while employing optimistic execution afterwards and smoothing as a recovery procedure in case execution would incorrectly predict the state of the cube. The latter algorithm projected the cube's position in the future, taking predicted latency effects into consideration, and optimistically provided this state to the simulated server and remote client. Smoothing or rollback could be used as recovery procedures in this optimistic algorithm, with rollback provided to demonstrate why it is not a good idea in optimistic programming for networked games. Details on these algorithms can be found in [6]; a screen shot of the simulation tool using client side adjustment of player state can be found in Figure 6.

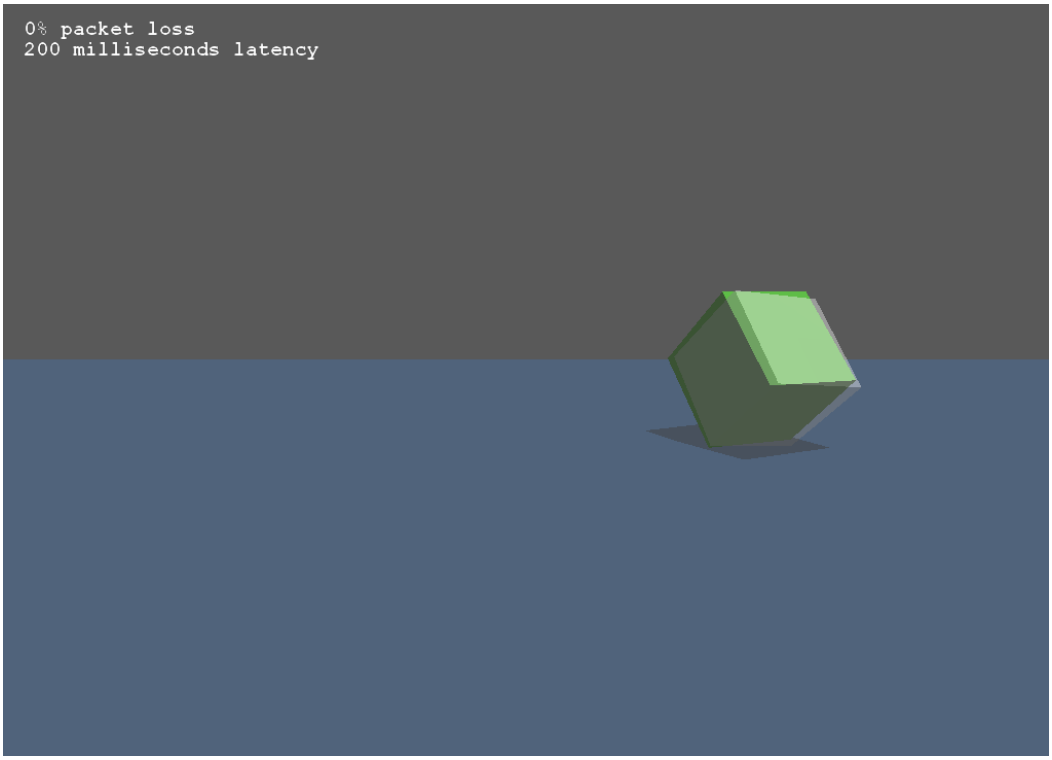

Figure 6: Client side adjustment of player state, with client state (green cube) matching server state (white cube) despite latency.

Discussion

Our experiences with implementing algorithms using primitives from New HOPE have demonstrated that New HOPE is sufficiently rich to represent a wide variety of optimistic algorithms suitable for use in networked video games. Clearly, however, more rigorous experimentation, including performance analyses, is necessary to ensure New HOPE is ready for widespread use.

We also learned that while it is possible to implement an Application Programming Interface (API) for New HOPE that presents optimism primitives (guesses, affirmations, denials, recoveries, padding, synchronization points, and so on) directly to the programmer in a fashion similar to [3], providing an API that hides these details would be quite difficult. To truly take advantage of optimism, the developers of networked games have to be aware of it and must be knowledgeable in using it wisely. This reaffirms the findings of [3].

This indicates a need for modelling tools to assist developers in reasoning about the desired levels of optimism in their games, the potential effects of optimism, and the kinds of guesses, padding, and recoveries required to support it. With the appropriate modelling tools, developing an optimistic networked game could be greatly simplified.

CONCLUDING REMARKS

Latency remains a challenging problem to the development and success of networked multiplayer games. New HOPE is aimed at reducing or eliminating the effects of latency to produce more enjoyable gaming experiences for players. To date, we have completed work in developing formalisms for New HOPE and developed algorithms using the optimism primitive from New HOPE as proof of concept, as discussed earlier in this paper. Furthermore, we have work well in progress towards a complete API and modelling tools for optimism in networked games. In the future, plans are to complete these developments, and to continue experimentation to validate New HOPE, as discussed above.

REFERENCES

1. Y. Bernier. "Latency Compensating Methods in Client/Server In-game Protocol Design and Optimization." *Presented at the 2001 Game Developers Conference*. San Francisco, California. March 2001.
2. J. Blow. Miscellaneous Rants. *Appeared in Game Developer Magazine*. May 2004.
3. C. Cowan. "A Programming Model for Optimism". PhD Thesis. Department of Computer Science, The University of Western Ontario. February 1995.
4. G. Fielder. "Zen of Networked Physics". *Presented at the 2004 Australian Game Developers Conference*. Melbourne, Australia, December 2004.
5. J. Fraser. *Zeroping Frequently Asked Questions*. Accessible online at: http://zeroping.home.att.net. April 2000.
6. R. Hanna. "Bringing New HOPE to Networked Games Research Project". *Undergraduate Thesis, Department of Computer Science, The University of Western Ontario. London, Ontario, Canada*, March 2005.
7. PricewaterhouseCoopers LLP. PricewaterhouseCoopers LLP Global Entertainment and Media Outlook: 2004-2008. *PWC Report*, 2004.