A Predictive Incremental Hierarchical Level of Detail Optimization Algorithm

Ashton E. W. Mason and Edwin H. Blake

CS99-04-00

Collaborative Visual Computing Laboratory
Department of Computer Science
University of Cape Town
Private Bag, RONDEBOSCH
7701 South Africa
e-mail: amason@cs.uct.ac.za, edwin@cs.uct.ac.za

Abstract

We present a new hierarchical level of detail optimization algorithm that is predictive and so may be used for active frame rate control. We base our approach on earlier work demonstrating the equivalence of level of detail optimization to the Multiple Choice Knapsack Problem (MCKP). We show that this equivalence is broken for hierarchical level of detail scene descriptions in which shared representations may be provided for groups of objects, and that the level of detail optimization problem for such descriptions is equivalent to a generalization of the MCKP which we call the Hierarchical MCKP. We present a greedy approximation algorithm for this Hierarchical MCKP whose solution we prove is guaranteed to be at least half-optimal for a useful subproblem in which more expensive selections provide diminishing returns. Furthermore we show that the typical behaviour of the algorithm is much better than half-optimal and that the instances in which it is not are relatively rare. The level of detail optimization algorithm we present is an incremental version of this greedy algorithm designed to exploit frame-to-frame coherence by basing its initial solution on the solution found for the previous frame. We prove the equivalence of the two algorithms by considering their state spaces and showing that both reach the same solution state.

1 Introduction

In order to guarantee consistent frame rates during interactive visualization, the dependency of the rendering complexity on the complexity of the visible scene must be eliminated. While advanced occlusion culling techniques and static level of detail selection methods based on perceptual importance serve to reduce the rendering complexity of typical frames, they tend to make the complexity of rendering dependent on the variable complexity of the visible scene rather than on the constant complexity of the entire model, thereby tending actually to *increase* the variability of frame rates. *Predictive* level of detail techniques promise a solution to this problem. They regulate frame rates by basing level of detail selection on estimates of available rendering resources, as well as on perceptual issues. Their aim is to select, for each frame, the set of detail levels that will provide the best overall perceptual benefit while limiting their total rendering cost to the available frame rendering time.

We propose a predictive hierarchical level of detail optimization algorithm that is a hybrid extension of those of Funkhouser and Séquin and Maciel and Shirley, and corrects problems noted with those algorithms. We develop a transformation from a hierarchical level of detail description to a non-hierarchical one that allows us to apply the constrained optimization approach of Funkhouser and Séquin to a hierarchical description. The advantage is that shared representations may be provided for groups of objects. This preserves the natural hierarchical description of scenes and saves additional rendering costs to allow better rendering of more important objects.

A contribution of this paper is to show that the hierarchical level of detail optimization problem is equivalent to a *hierarchical generalization* of the Multiple Choice Knapsack Problem (MCKP). Our algorithm is essentially a greedy approximation algorithm for this problem, and is a hierarchical extension of a greedy algorithm for the MCKP that we present in [7]. Our algorithm corrects a problem with the algorithm of Funkhouser and Séquin and makes use of a new metric, *relative value*, that measures the desirability of potential selections. The result is that our algorithm's solution is always at least half as good as the optimal solution, for a useful subproblem of the Hierarchical MCKP. We show that its solution is typically much better.

The remainder of the paper is organized as follows. In Section 2 we review related work. In Section 3 we present a formal definition of a generalized hierarchical level of detail description. In Section 4 we introduce the Hierarchical MCKP. In Section 5 we present our greedy algorithm for the Hierarchical MCKP. In Section 6 we prove the guaranteed half-optimality of the greedy algorithm's solution. In Section 7 we present our predictive hierarchical level of detail optimization algorithm. In Section 8 we discuss the limitations and advantages of our algorithms. Finally in Section 9 we offer some concluding remarks.

2 Background

Level of detail rendering is founded on the provision of multiple drawable representations, or *impostors*, for scene objects at a range of detail levels [1]. Level of detail techniques have traditionally been used to reduce the complexity of rendering, and hence improve frame rates, by reducing the rendering of detail made invisible by perspective projection. Funkhouser and Séquin [3] were among the first to note that level of detail could be used not only to opportunistically *reduce* the complexity of rendering but also to *limit* it entirely by actively selecting only as much detail as there is time to render. They refer to this as *predictive* level of detail optimization.

Funkhouser and Séquin note the equivalence of level of detail optimization to the *Multiple Choice Knapsack Problem* (MCKP), a variation of the well-known NP-complete Knapsack Problem [6]. The Knapsack Problem models the general situation in which a cost-limited subset of a set of *candidate items* with various *costs* and *profits* must be selected so as to maximize their total profit. In MCKP the items are partitioned into various types or *candidate subsets*, and exactly one item must be selected from each type. The MCKP is defined as follows:

Definition 1 *The Multiple Choice Knapsack Problem*

Given a set N of n candidate items, *a partition into disjoint* candidate subsets $N_1, \ldots, N_r$ *of the item set N and a* knapsack, *with*

$$\begin{aligned} p_j &= \text{profit } \textit{of item } j && (1)\\ w_j &= \text{cost } \textit{of item } j && (2)\\ c &= \text{capacity } \textit{of the knapsack} && (3)\end{aligned}$$

maximize

$$z = \sum_{j=1}^{n} p_j x_j \tag{4}$$

subject to

$$\sum_{j=1}^{n} w_j x_j \leq c \tag{5}$$

$$\sum_{j \in N_k} x_j = 1 \quad \forall \quad k \in \{1, \ldots, r\} \tag{6}$$

$$x_j \in \{0, 1\} \quad \forall \quad j \in N \tag{7}$$

$$N = \{1, \ldots, n\} = \bigcup_{k=1}^{r} N_k \tag{8}$$

assuming

$$N_h \cap N_k = \emptyset \ \forall \ h \neq k. \tag{9}$$

The correspondence of MCKP to rendering is explained by noting that in predictive level of detail optimization a rendering-cost-limited subset of a set of available object representations must be selected so as to maximize their total perceptual benefit, with the additional requirement that exactly one representation must be selected for each visible object. Funkhouser and Séquin propose the use of *benefit* and *cost* heuristics that provide rough predictions of the perceptual benefit (profit) and rendering cost (cost) of potential object representations.

Funkhouser and Séquin describe a greedy approximation algorithm for MCKP[1] which considers the candidate items in descending order of *value* (profit / cost), selecting each item if its selection can be afforded. If it occurs that the item under consideration belongs to the same candidate subset as an item already in the knapsack, the algorithm retains whichever item has greater *profit*, and discards the other [3]. The complexity of the algorithm is $O(n \log n)$. Funkhouser and Séquin claim that its solution is at least half-optimal in the worst case. However their algorithm is flawed and its worst case solution is arbitrarily bad. Consider the counterexample in which there are 7 items with profits $\overline{p} = (10, 900, 910, 10, 600, 10, 400)$ and costs $\overline{w} = (1, 100, 700, 1, 500, 1, 400)$,

[1]Funkhouser and Séquin actually state that level of detail optimization is equivalent to the *Continuous Multiple Choice Knapsack Problem*, a relaxation of MCKP in which items may be fractionally selected [6] [4]. In this sense their algorithm is an algorithm for C(MCKP), rather than MCKP. However their algorithm never selects fractional portions of items and is therefore just as much an algorithm for MCKP. Moreover, in the level of detail optimization problem posed by Funkhouser and Séquin there is no concept of rendering only part of an object representation.

partitioned into $r = 3$ candidate subsets. Candidate subset N_1 contains items 1, 2 and 3, N_2 contains items 4 and 5, and N_3 contain items 6 and 7. The capacity of the knapsack is $c = 1000$. The solution reached by the algorithm in this instance is $\overline{x} = (0, 0, 1, 1, 0, 1, 0)$ with total profit $z^g = 930$, while the optimal solution is $\overline{x} = (0, 1, 0, 0, 1, 0, 1)$, with total profit $z = 1900$. Therefore the algorithm's solution to this instance is less than half as good as the optimal solution.[2]

Rather than perform a complete greedy approximation for each frame, Funkhouser and Séquin formulate an incremental level of detail optimization algorithm, based on their greedy algorithm, that accepts as input an initial solution derived from the previous frame. The complexity of the incremental algorithm is still $O(n \log n)$ in the worst case but the average case efficiency is improved by exploiting frame-to-frame coherence.

A major limitation of the Funkhouser and Séquin algorithm is that, being founded on the MCKP, it is inherently non-hierarchical and is incapable of dealing with the shared representations for groups of objects that characterize hierarchical level of detail scene descriptions. Shared object representations afford many benefits such as the ability to provide coherent and consistent low detail representations for groups of related objects and the additional savings in rendering and optimization costs that this allows.

Maciel and Shirley [5] present a hierarchical predictive level of detail optimization algorithm that is a hierarchical extension of the predictive approach of Funkhouser and Séquin. However their algorithm represents a naive attempt to extend the already flawed MCKP greedy algorithm of Funkhouser and Séquin to the hierarchical level of detail optimization problem, and provides no guarantees of solution quality. Furthermore it is non-incremental and must perform a complete greedy approximation for each frame.

Our level of detail algorithm represents both a correction of the flaws afflicting the Funkhouser and Séquin algorithm and a correct extension of their predictive incremental approach to hierarchical level of detail descriptions with shared object representations. This paper is based in part on [8], in which we describe an earlier algorithm for the same problem. Here we correct an error in that algorithm's detail selection heuristic that caused its solution to be less than half-optimal in the worst case. In addition we provide a formal proof of the correctness of our algorithm.

3 Hierarchical Level of Detail Description

In this section we present a formal definition of a generalized hierarchical level of detail scene description that will serve as the basis for the rest of the paper. Our definition is general and makes no assumptions about practical implementation. The characterizing feature of this description is that multiple shared representations may be provided for groups of related scene objects. An *object* is defined recursively as consisting of other objects that are its children or *parts*. The entire scene is represented by a hierarchy of such objects whose root is called the *scene object*. Each object may optionally be provided with a set of *impostors*, or drawable representations, that represent it and therefore all of its parts. The leaf objects must each be provided with at least one impostor. Where multiple impostors are associated with a single object, they are ordered uniquely according to increasing detail. The impostors of the parts of objects together form more detailed representations of those objects. Figure 1 shows an example of a simple level of detail hierarchy.

We define a formal hierarchical generalization of the concept of a *level of detail*. Objects have multiple hierarchically defined levels of detail consisting of both their own *explicit* impostor representations and the *implicit* representations consisting of the combinations of the impostors of their descendants. Each level of detail corresponds to a unique set of selected impostors:

[2] Note that the counterexample is valid for C(MCKP) as well, since C(MCKP) is a relaxation of MCKP.

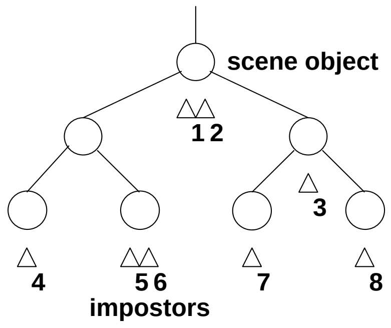

Figure 1: **A simple level of detail hierarchy.** Objects are represented by circles, and their impostors by triangles. The multiple impostors of each object are shown in order of increasing detail from left to right. Impostors are numbered for convenience.

Definition 2 *Level of Detail*
A level of detail s *of an object* O *is a set of impostors* $\{i_1, i_2, i_3, \ldots, i_n\}$. *The impostors* $i_1, i_2, i_3, \ldots, i_n$ *are selected such that exactly one of the impostors on the path from* O *to each of the leaves of the subtree rooted at* O *is an element of* s.

For example a valid level of detail of the scene object in Figure 1 is the set of impostors $\{4, 6, 3\}$. This definition ensures that each level of detail of an object provides some complete and unambiguous representation of every part of that object; be it one of its associated impostors or a subset of the impostors of its descendants. In addition objects that are parts of other objects may also be represented by the impostors of their ancestors.

We define the *replacement set* of an impostor to refer to the set of impostors that constitute the immediately higher detail representation of the object that owns that impostor:

Definition 3 *Replacement Set*
The replacement set *of an impostor* i *belonging to an object* O *is:*

1. *The immediately higher detail impostor of* O*, if one exists.*
2. *The set of the lowest detail impostors of the nearest impostor-bearing descendants of* O*, otherwise.*

For example the replacement set of impostor 2 in Figure 1 is $\{4, 5, 3\}$. The highest detail impostors of leaf objects have no replacement sets. We define an *incrementation* of a level of detail s of an object O to be the replacement of some impostor $i \in s$ by its replacement set R. Conversely a *decrementation* of s is the replacement of some complete replacement set $R \subseteq s$ by the impostor i whose replacement set is R. In general a level of detail s may be incremented and decremented in many different ways, where each corresponds to the replacement of a different impostor or replacement set in s. The levels of detail of each object are partially ordered by the following relation:

Definition 4 *Partial Ordering of Levels of Detail*
Two levels of detail s *and* t *of an object* O *are related by* $s \leq t$ *if there exist levels of detail* $l_1, l_2, l_3, \ldots, l_n$ *such that* $l_1 = s$, $l_n = t$, *and* l_{i+1} *is the result of some incrementation of* l_i *for all* $i \in \{1, 2, 3, \ldots, n-1\}$.

If $s \leq t$ and $s \neq t$ then we say that s is a *strictly lower* level of detail of O than t. The *lowest* and *highest* levels of detail of an object are those such that there exist no levels of detail that are strictly lower and strictly higher, respectively.

Lastly we define two terms that reflect a partial ordering of replacement sets:

Definition 5 *Ancestor Replacement Sets*

We say that a replacement set R is an ancestor replacement set *of another replacement set S if there exists a (possibly empty) list of replacement sets $R_1, R_2, R_3, \ldots, R_n$ such that $R = R_1$, $S = R_n$, and R_{i+1} is the replacement set of some impostor in R_i for $i \in \{1, 2, 3, \ldots, n-1\}$.*

Definition 6 *Descendant Replacement Sets*

S is a descendant replacement set *of R if R is an* ancestor replacement set *of S.*

In the example shown in Figure 1, $\{3, 4, 5\}$ is a descendant replacement set of $\{2\}$ and an ancestor replacement set of $\{6\}$ and $\{7, 8\}$. Note that all replacement sets are trivially ancestors and descendants of themselves.

4 Hierarchical Multiple Choice Knapsack Problem

In the hierarchical level of detail scene description defined in Section 3, each group (or non-leaf) object is the union of its parts, or children. Therefore impostors of group objects are essentially shared representations of all of the parts of those group objects. By our definition they function as lower detail representations of those parts than any of the impostors that are explicitly associated with the parts themselves. We may therefore redraw the hierarchy equivalently by transforming group object impostors into shared low-detail impostors of their children, as long as we note that the shared impostors are constrained and must be selected in unison for all of the parts, if at all. By repeatedly applying this transformation we may create an "empty" or flat hierarchy with impostors only at the leaves (see Figure 2). Each leaf has as its impostors all of its own impostors plus a series of lower detail impostors inherited in top-down order from its ancestors in the hierarchy. The equivalence is subject to a set of *constraints*, one for each original group impostor: the leaf objects that share each inherited group impostor *must take on that shared impostor in unison*. The resulting flat hierarchy is essentially a hierarchically constrained non-hierarchical level of detail description, exactly equivalent to the original hierarchical one. The immediately higher impostors of the objects that share each inherited group impostor together constitute the replacement set of that impostor.

We can now show that the hierarchical level of detail optimization problem is equivalent to a hierarchical generalization of the Multiple Choice Knapsack Problem, shown in Figure 3. In this *Hierarchical MCKP*, the candidate subsets are not disjoint and some candidate items are shared between multiple candidate subsets. Each candidate subset corresponds to an object in the constrained non-hierarchical description, and its candidate items correspond to the impostors of the object.

The definition of the Hierarchical MCKP is identical to that of the MCKP given in Section 2 except that the candidate subsets are not necessarily disjoint. Line (9) is replaced with a stipulation that the *root item* $o \in N_k$, $k = 1, \ldots, r$ has a *replacement set* R_o where the replacement set of an item i is a set of items $R_i = \{i_1, i_2, i_3, \ldots, i_{z_i}\}$ such that:

1. For each candidate subset N_k of which i is an element, there exists exactly one item $j \in R_i$ that is an element of N_k.

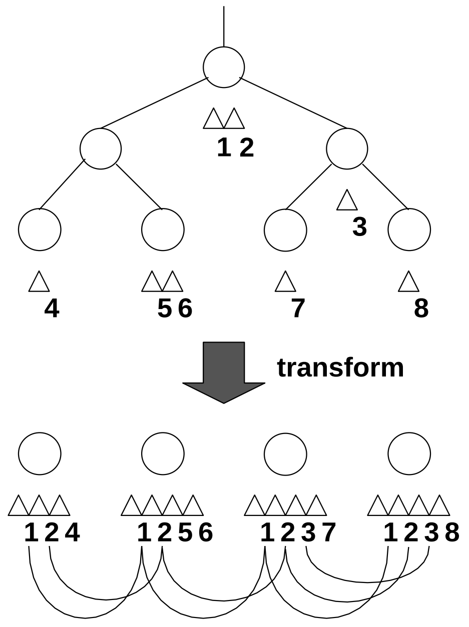

Figure 2: **Transformation of a simple level of detail hierarchy to its equivalent constrained non-hierarchical description.** The impostors of group objects have all been transformed into shared impostors of the leaf objects. The constraints between shared impostors, shown as links, imply that the objects which share those impostors must take them on in unison.

2. Each item $j \in R_i$ may or may not have a replacement set.
3. All replacement sets are mutually disjoint.

5 Greedy Algorithm for the Hierarchical MCKP

In this section we present our greedy algorithm for the Hierarchical MCKP. The algorithm accepts as input an instance of the Hierarchical MCKP and produces as output a feasible solution to that instance. The algorithm begins with the simplest feasible solution and iteratively replaces items with their replacements sets as far as the available cost will allow. It maintains the feasibility of the solution by always replacing an item with its complete replacement set and ensuring that replacement sets are only considered for selection when the items they replace have already been selected. It maximizes the quality of the solution by favouring, when given the choice, replacements that result in the greatest increase in profit for the smallest increase in cost. In order to determine the most desirable replacements, the algorithm makes use of a simple selection heuristic based on a metric *relative value* that measures the profit density of replacement sets relative to the items that they replace:

Definition 7 *Relative Value*

The relative value *of the replacement set R of an item i is measured with respect to i and is defined as follows:*

$$\textit{relative value}(R) = \frac{(\sum_{j \in R} p_j) - p_i}{(\sum_{j \in R} w_j) - w_i}$$

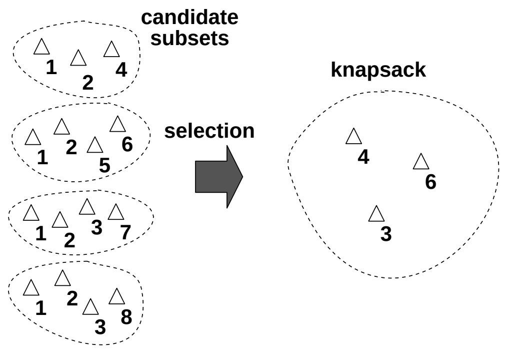

Figure 3: **The Hierarchical Multiple Choice Knapsack Problem.** The items correspond to the impostors of objects in the non-hierarchical level of detail description. They are divided into subsets corresponding to the objects to which the impostors belong, and items that correspond to shared impostors are members of more than one subset. Only one item may be selected from each subset.

The relative value of replacement set R provides a measure of the advantage gained by using it to replace the item i that it replaces, measured as the ratio of the differences in total profit and cost between the replacement set R and the replaced item i.

The algorithm maintains a list of replacement sets currently available for selection, ordered by descending relative value. It initially selects the imaginary root item (See Section 4) that has no profit and cost and is an element of every candidate subset, and inserts into the replacement set list the replacement set of this item. It then greedily considers the remaining replacement sets in order of descending relative value, using the replacement set list to ensure that each replacement set is only considered after the item it replaces has been selected. While the replacement set list is not empty, the algorithm considers the first replacement set in the list for replacement, substituting it for the item it replaces if this replacement can be afforded. If the replacement is made then the replacement set is removed from the replacement set list and the replacement sets (if any) of the items it contains are inserted into the list in descending relative value order. Otherwise the replacement set is simply removed from the list and discarded. If it is the first replacement set to be so discarded it is marked as the *critical replacement set*. At any stage the replacement sets that are available in the replacement set list are those whose associated items are currently selected, and they are considered in descending order of relative value.

When this greedy selection terminates (due to the replacement set list being found to be empty) the solution reached is compared against the lowest cost feasible solution containing the critical replacement set. If this *critical replacement set solution* has greater profit than the greedy selection and has total cost less than or equal to the size of the knapsack then it is selected instead.

Finding the critical replacement set is simple: it is found as a by-product of the greedy selection stage. Finding the critical replacement set solution however requires an iterative selection process similar to the greedy selection stage but taking no notice of relative value orderings and instead selecting only replacement sets that represent candidate subsets that are also represented by the critical replacement set.

6 Proof of Half-Optimality

In this section we prove the half-optimality of the greedy algorithm described in Section 5. Our algorithm's solution is at least half-optimal for instances of the Hierarchical MCKP in which the replacement set of an item always has *greater total cost and profit* than that item and *lower relative value* than all of its ancestors (if any). This implies that more expensive selections should always provide *diminishing returns*. We assume here that those requirements are satisfied.

The approach taken in the proof is to formulate an expression relating the profit of the optimal solution to the profit of the intermediate solution reached by the greedy algorithm immediately before the consideration and rejection of the critical replacement set. At that point we know, by virtue of the selection heuristic and the diminishing returns assumption (Section 5), that all of the replacement sets selected have greater relative value than those not selected. Using this we show that the maximum error of the algorithm is bounded by the profit of the critical replacement set. Recalling that the algorithm also considers the critical replacement set solution, we deduce that the algorithm's solution is at least half-optimal.

Given an instance of the Hierarchical MCKP, let the profit of the optimal solution to this instance be z. Let G be the set of items in the intermediate solution reached by the greedy algorithm immediately before the critical replacement set is considered (and rejected), and let $z^g = \sum_{i \in G} p_i$ be the profit of this intermediate greedy solution.[3]

Therefore

$$z = z^g + \sum_{j \in A}((\sum_{i \in R_j} p_i) - p_{r_j}) - \sum_{j \in B}((\sum_{i \in R_j} p_i) - p_{r_j}) \tag{10}$$

where r_j is the item whose replacement set is R_j, A is the set of replacement sets that would be selected in the process of selecting the optimal solution but were not selected in the process of selecting G (those where the algorithm has "underselected"), and B is the set of those replacement sets that were selected in the process of selecting G but would not be selected in the selection of the optimal solution (where the algorithm has "overselected").

In this step we consider the replacement sets that are in A. When the critical replacement set S was considered (and rejected) the set of currently selected items was exactly G. Therefore the critical replacement set was considered as the replacement for some item t that is an element of G, and was considered instead of some replacement set V that is the replacement set of some item $v \in G$ and is an ancestor replacement set of R_j. This implies that V has lower relative value (with respect to v) than S (with respect to t):

$$\frac{(\sum_{i \in V} p_i) - p_v}{(\sum_{i \in V} w_i) - w_v} \leq \frac{(\sum_{i \in S} p_i) - p_t}{(\sum_{i \in S} w_i) - w_t}.$$

Now because the replacement sets of items always have lower relative value than the replacement sets containing those items (by assumption), all of the replacement sets in A must have lower relative value than S:

$$\frac{(\sum_{i \in R_j} p_i) - p_{r_j}}{(\sum_{i \in R_j} w_i) - w_{r_j}} \leq \frac{(\sum_{i \in S} p_i) - p_t}{(\sum_{i \in S} w_i) - w_t} \quad \forall \quad j \in A. \tag{11}$$

In this step we consider the replacement sets that are in B. When the critical replacement set S was considered for selection (and rejected) the set of currently selected items was exactly G.

[3] In practice the algorithm may also select other later replacement sets, replacing items in G, but since every replacement increases the total profit of the selected items (by assumption), we know that the profit of the final greedy solution is greater than or equal to z_g.

There must therefore exist a list of replacement sets $J_1, J_2, J_3, \ldots, J_z$ such that $J_1 = R_j$, $J_z \subseteq G$, and J_{i+1} is the replacement set of some item in J_i for all of $i = 1, 2, 3, \ldots, z-1$.

Likewise there also exists a list of replacement sets $M_1, M_2, M_3, \ldots, M_y$ where M_1 is the replacement set of some item in the cheapest feasible solution, $M_y = S$ and M_{i+1} the replacement set of some item m_i in M_i for all of $i = 1, 2, 3, \ldots, y-2$. Note that M_y is not selected as the replacement for some item in M_{y-1}, because M_y (ie. S) is not selected at all.

Then we know that the algorithm at some stage replaced some item j_{z-1} in J_{z-1} with J_z instead of replacing some item m_u in M_u with M_{u+1}, for some $u \in \{1, 2, 3, \ldots, y-1\}$, since J_z was selected and S was not. Therefore

$$\frac{(\sum_{i \in J_z} p_i) - p_{j_{z-1}}}{(\sum_{i \in J_z} w_i) - w_{j_{z-1}}} \geq \frac{(\sum_{i \in M_{u+1}} p_i) - p_{m_u}}{(\sum_{i \in M_{u+1}} w_i) - w_{m_u}}.$$

Now because the replacement sets of items always have lower relative value than the replacement sets containing those items (by assumption), all of the replacement sets in B must have greater relative value than S:

$$\frac{(\sum_{i \in R_j} p_i) - p_{r_j}}{(\sum_{i \in R_j} w_i) - w_{r_j}} \geq \frac{(\sum_{i \in S} p_i) - p_t}{(\sum_{i \in S} w_i) - w_t} \quad \forall \quad j \in B. \tag{12}$$

Therefore, from (10), (11) and (12) we have

$$\begin{aligned} z &\leq z^g + \sum_{j \in A}((\sum_{i \in R_j} w_i) - w_{r_j})\frac{(\sum_{i \in S} p_i) - p_t}{(\sum_{i \in S} w_i) - w_t} - \sum_{j \in B}((\sum_{i \in R_j} w_i) - w_{r_j})\frac{(\sum_{i \in S} p_i) - p_t}{(\sum_{i \in S} w_i) - w_t} \\ &\leq z^g + \left[\sum_{j \in A}((\sum_{i \in R_j} w_i) - w_{r_j}) - \sum_{j \in B}((\sum_{i \in R_j} w_i) - w_{r_j})\right]\frac{(\sum_{i \in S} p_i) - p_t}{(\sum_{i \in S} w_i) - w_t}. \qquad (13) \end{aligned}$$

Let $\bar{c} = c - \sum_{i \in G} w_i$ be the space left in the knapsack after the selection of G, immediately before the rejection of the critical replacement set S. From the fact that S was rejected we know that the difference in cost between S and t is greater than $\bar{c}$:

$$\bar{c} \quad < \quad (\sum_{i \in S} w_i) - w_t. \tag{14}$$

Furthermore we know that the total difference in cost between the optimal solution and the intermediate greedy solution G must be less than or equal to $\bar{c}$:

$$\sum_{j \in A}((\sum_{i \in R_j} w_i) - w_{r_j}) - \sum_{j \in B}((\sum_{i \in R_j} w_i) - w_{r_j}) \quad \leq \quad \bar{c}.$$

Therefore, from (14),

$$\sum_{j \in A}((\sum_{i \in R_j} w_i) - w_{r_j}) - \sum_{j \in B}((\sum_{i \in R_j} w_i) - w_{r_j}) \quad \leq \quad (\sum_{i \in S} w_i) - w_t \tag{15}$$

and so, from (13) and (15),

$$\begin{aligned} z &\leq z^g + ((\sum_{i \in S} w_i) - w_t)\frac{(\sum_{i \in S} p_i) - p_t}{(\sum_{i \in S} w_i) - w_t} && (16) \\ &\leq z^g + (\sum_{i \in S} p_i) - p_t && (17) \\ &\leq z^g + \sum_{i \in S} p_i. && (18) \end{aligned}$$

Recall that the greedy algorithm compares the total profit of the final greedy solution (which is greater than or equal to z^g) to the total profit z^s of the cheapest solution containing the critical item, and keeps whichever solution is better. That is, the algorithm's solution has profit $z^h \geq \max(z^g, z^s)$. Clearly $z^s > \sum_{i \in S} p_i$, so $z^h \geq \max(z^g, \sum_{i \in S} p_i)$. Therefore, from (18),

$$z^h \geq \frac{1}{2} z$$

and the profit of the algorithm's solution is guaranteed to be at least half the profit of the optimal solution.

7 Hierarchical Level of Detail Optimization Algorithm

Our incremental hierarchical level of detail optimization algorithm, shown in Figure 4, is an equivalent incremental version of the Hierarchical MCKP greedy algorithm described in Section 5. Its advantage over that algorithm is purely one of efficiency: it exploits frame-to-frame coherence by basing its initial solution on the solution found for the previous frame.

```
begin
    set $L \leftarrow$ the initial solution
    set done $\leftarrow$ FALSE
    while done = FALSE {
                // increment $L$, if we can
        if $L$ is not the highest level of detail then {
            find $i$, the impostor in $L$ whose replacement set has highest relative value
            set $R \leftarrow$ the replacement set of $i$
            set $L \leftarrow (L - \{i\}) \cup R$
        }
                // decrement $L$, while the total rendering cost is too high
        while $\sum_{i \in L}$ Cost$(i) >$ rendering cost limit {
            find $S \subseteq L$, the replacement set in $L$ with the lowest relative value
            set $j \leftarrow$ the impostor whose replacement set is $S$
            set $L \leftarrow (L - S) \cup \{j\}$
            if $S = R$ then set done $\leftarrow$ TRUE
        }
    }
end
```

Figure 4: **The incremental hierarchical level of detail optimization algorithm.**

The algorithm is applied once per frame and its output is a level of detail of the scene object for that frame. Its input is the level of detail selected for the previous frame[4] and a constant *rendering cost limit* that represents the rendering time available for this frame. The algorithm guarantees that the total predicted rendering cost of the selected level of detail is lower than the rendering cost limit. In addition it attempts to maximize the total predicted perceptual benefit (or profit) of the selection.

[4] Or any valid level of detail, in the case of the first frame.

The algorithm is iterative, repeatedly incrementing and decrementing the selected level of detail until the final solution is found. In each iteration the selected level of detail is incremented once, then decremented repeatedly while the total rendering cost is greater than the rendering cost limit. Recall that a level of detail may in general be incremented and decremented in many different ways. The incrementation selected in each step is that which replaces the currently selected impostor whose *replacement set* has the *highest relative value*. Conversely the decrementation selected is that which deselects the *currently completely selected* replacement set with the *lowest relative value*.

The repeated iteration terminates when the incrementation and decrementation operations of the same iteration select and deselect the same replacement set. When this occurs there is no further work for the algorithm to do. After termination of the algorithm we render the impostors constituting the selected level of detail. Note that the algorithm as described does not consider the critical replacement set solution (Section 5). The critical replacement set is that which is both selected and deselected in the final iteration of the algorithm. However we feel that in practical level of detail situations the pathology in which the critical replacement set contributes significantly to the final solution is unlikely to arise.

The incremental algorithm is equivalent to the original greedy algorithm as long as the diminishing returns assumption holds (see Section 5). Note that the incrementation operation is identical to the selection heuristic of the greedy algorithm. Decrementation is the inverse of incrementation.

8 Discussion

Note that the maximum error of the greedy algorithm is bounded by the difference in profit between the critical replacement set and the item it replaces (see equation (17)). Therefore as the granularity of the candidate items with respect to the knapsack becomes finer, the maximum error of the algorithm tends to zero. In practical level of detail applications the performance of the algorithm can be expected to be much better than half-optimal.[5]

Recall that the algorithm depends on the diminishing returns assumption for its half-optimality. This assumption implies that higher detail representations of objects must provide increased perceptual benefit at the expense of increased rendering cost, with diminishing returns for increasingly more detailed representations. These requirements are likely to be satisfied in most real-world applications. Level of detail problems in which more expensive renderings do not provide diminishing returns are uncommon. For example the successive addition of more detail to a polygonal mesh model generally results in progressively smaller improvements in visual perception. Therefore the algorithm can be expected to perform well in most practical level of detail applications.

The complexity of the greedy algorithm is $O(n \log n)$. The number of replacements made is $O(n)$ in the number of candidate items (and hence in the number of replacement sets). Each replacement involves the deletion of the replacement set at the head of the list, which is $O(1)$, and the in-order insertion of (we assume) $O(1)$ new replacement sets, each at the expense of $O(\log n)$ complexity. The complexity of the greedy selection stage is therefore $O(n \log n)$. After the greedy selection stage the critical item solution must be found. The complexity of this is $O(n)$. The entire algorithm is therefore $O(n \log n)$.

The worst-case complexity of the incremental algorithm is also $O(n \log n)$. However by exploiting frame-to-frame coherence it generally avoids performing a full computation for every frame and instead only computes the difference between each frame and the last. Therefore its

[5]A similar observation is made for other Knapsack Problem heuristics in [2].

average case complexity is significantly lower and its best case is $O(1)$. In [9] we present an experimental evaluation of the algorithm that suggests its average complexity is closer to O(n).

9 Conclusion

We have presented a predictive hierarchical level of detail optimization algorithm. Our algorithm is truly hierarchical and allows the use of hierarchical scene descriptions with shared representations for groups of objects. The benefits of this are that consistent and coherent shared representations may be provided for groups of related objects, saving rendering and optimization costs and preserving the completeness of the scene representation even in demanding rendering situations. Because of its predictive nature our algorithm guarantees constant frame rendering times by ensuring that the predicted rendering cost of the selected scene representation is always lower than the available rendering time. Furthermore it actively optimizes the perceptual benefit of the selected scene representation and its solution has guaranteed levels of predicted perceptual quality. By virtue of this we correct problems with previous algorithms that caused their solutions to be arbitrarily bad. Lastly our algorithm is incremental and so exploits frame-to-frame coherence for improved efficiency by basing its initial solution on the approximate solution found for the previous frame.

References

[1] J. Clark. Hierarchical geometric models for visible surface algorithms. *Communications of the ACM*, 19(10):547–554, 1976.

[2] M. L. Fisher. Worst-case analysis of heuristic algorithms. *Management Science*, 26(1):1–17, January 1980.

[3] T. A. Funkhouser and C. H. Séquin. Adaptive display algorithm for interactive frame rates during visualization of complex virtual environments. In *Computer Graphics Proceedings Annual Conference Series*, volume 27, pages 247–254. ACM SIGGRAPH, August 1993.

[4] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[5] P. W. C. Maciel and P. Shirley. Visual navigation of large environments using textured clusters. In *1995 Symposium on Interactive 3D Graphics*, pages 95–102, April 1995.

[6] S. Martello and P. Toth. *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley and Sons Ltd., 1990.

[7] A. Mason and E. Blake. An incremental greedy approximation algorithm for the multiple choice knapsack problem. Technical Report CS99-03-00, Department of Computer Science, University of Cape Town, 1999.

[8] A. E. W. Mason and E. H. Blake. Automatic hierarchical level of detail optimization in computer animation. In D. Fellner and L. Szirmay-Kalos, editors, *Computer Graphics Forum, proceedings of Eurographics '97*, volume 16, pages 191–199. Eurographics, Blackwell Publishers, 1997.

[9] S. Nirenstein, S. Winberg, A. Mason, and E. Blake. Hierarchical level of detail optimization for rendering of radiosity scenes. Technical Report CS99-02-00, Department of Computer Science, University of Cape Town, 1999.