



UXREROLLING **(brand-new rerolling algorithm)**



Toshihiro KONDA

Aim

- ▶ **UXREROLLING**

- ▶ Consider optimal loop-rerolling algorithm

- ▶ We expect the following effects

- ▶ Maximum use of instruction cache (I\$)

- ▶ Promote loop vectorization

- ▶ legacy code performance improvement



Background

- ▶ Generally known “**loop rerolling**” algorithm
 - ▶ It can only reroll relatively simple loops.

- ▶ Why ?
 - ▶ With existing technology,
 - ▶ difficult to apply to complex loops
 - ▶ need a complicated logic
 - ▶ etc.



Solution 1. (Premise)

- ▶ **Unrolled code . . .**

- ▶ Regardless of manual, automatic.

- ▶ Number of unrolls is obtained by applying Greatest Common Divisor algorithm (GCD) to the following

- ▶ Increment value of induction variable,
 - ▶ Index indicating the position of the array,
 - ▶ Array type



Solution 2. (Preparation)

- ▶ **From unrolled code, generate an operation tree representing a chain of operations**
- ▶ **A fixed algorithm that does not apply deformation (tree height reduction, etc.) is used for generating the operation tree**
 - ▶ **As a result, it is possible to deform while maintaining the original structure**



Solution 3. (Preparation)

- ▶ **A depth-first search is performed on the operation tree and it is taken as “operation character string”**
- ▶ **Whether to trace either the left subtree or the right subtree is fixed, which is OK.**
 - ▶ **Uniqueness is preserved.**
- ▶ **Uniqueness can be guaranteed by including (mainly) characteristics of load instructions.**



Solution 4. (“rerolling possibility”)

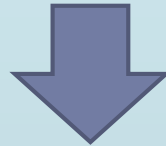
- ▶ It is possible to calculate similarity with three parameters of
 - ▶ Longest Common Subsequence (LCS),
 - ▶ Edit distance, and
 - ▶ String length
- ▶ among plural **“character strings”** that can be created.
- ▶ e.g.
 - ▶ When “DEBFGCA”, “BFGCA”, and “BCA” are given,
 - ▶ When calculating the LCS length starting from the shortest **“character string”**, the three LCS lengths agree.
 - ▶ And editing distance is different but editing to the same **“character string”** is possible only by inserting.



Solution 5. (“rerolling possibility”)

▶ Definition

- ▶ Common LCS length
- ▶ It is possible to "edit" the same "**character string**" by just inserting



- ▶ Define this group to be **rerollable**.

▶ Similarly, if there are several groups of the same number of “**character strings**”

- ▶ It is judged whether to have “**rerolling possibility**” or not by considering calculation of induction variables, that all are “rerollable”.



Solution 6. (“loop-rerollable”)

► Definition

- When there was “**rerolling possibility**”,
 - Depending on the index of the array operated by either load instructions or store instructions (store easy),
 - Make sure that all pairs with “**rerolling possibility**” are accessing the array with the same increment value ratio (depending on the type of variable).
- When this is possible, it is defined as “**loop-rerollable**”.



Solution 7. (loop rerolling process)

- ▶ **"Character string"** is made redundant (common) by **"insertion operation only"** out of the algorithms for calculating the edit distance.
 - ▶ In order to reproduce the original operation in the case of the operation tree from which the common expression is removed.
- ▶ When performing the insertion operation, it is necessary to keep the shape of the operation tree in order to guarantee the uniqueness.
 - ▶ Therefore, we add the shape information of the tree and make the operation tree **"character string"**.



Solution 8. (loop rerolling process)

- ▶ When loops are “**loop-rerollable**”,
 - ▶ It is possible to reproduce the operation tree before applying loop unrolling based on shape information.
 - ▶ If the shape can be reproduced, since loops are “**loop-rerollable**” afterward, loop rerolling can actually be performed.
- ▶ This is the essence of the algorithm.



Additional functions

- ▶ **When the operation is included in the tree "only" other than the largest operation tree,**
 - ▶ with predicate, or ternary operator, it is possible to do more than simply loop rerolling.
- ▶ It is also easier by treating loops as “**character strings**”.



Points to keep in mind

- ▶ **Disadvantages to be expected**
 - ▶ **Depending on**
 - ▶ **Number of loops to iterate and**
 - ▶ **SIMD width,**
- ▶ **there is a possibility that the effect when common expression is taken may become larger.**
- ▶ **In order to prevent it, it is necessary to adopt a good estimate beforehand.**



Conclusion

- ▶ **With conventionally known methods, loop rerolling in complicated cases is quite difficult.**
- ▶ **By this method . . .**
 - ▶ By analyzing the similarity of operations over a plurality of groups, "**easy**" loop rerolling can be applied.
- ▶ **At the same time the original purpose is also achieved**
 - ▶ **Maximum use of instruction cache (I\$)**
 - ▶ **Promote loop vectorization**
 - ▶ **legacy code performance improvement**



Thanks.

