Reading: DSO ch. 7, 8, 9 and 10.

Maze (graph) search/traversal, and Queues in Discrete Event Simulation

as 4 views of the SAME IDEA (context-free language) Expressions

Recursion, Trees, Stacks, and

And half of this course covers:

CSI 310: Lecture 18
We will review the 2 easy ways to make recursion:

1. "Faith"
2. Mathematical Induction, careful expression of what programs should do, best

We will review the basic terms, rules, and techniques to solve recursive problems.

Algorithm:

(1) Understanding data structures, nested documents/objects, and

(2) Technique to solve problems.

(3) Removable Bligeance.

Examples: fact(n), mergeSort(char**, int n), your recursion.

Definition: A function is recursive if the body of the function sometimes

ree, recursion, expressions, stacks are closely related.

University at Albany Computer Science Dept.
CALLS.

*ONLY think about the RESULTS of DIRECT RECURSIVE
*activities—YOU DO NOT WORRY about activities during recursive

Then, you verify PRE-CONDITIONS, then

The "right thing" means after YOU verify PRE-CONDITIONS, then

The "right thing" does the "right thing"

Way 1: Just figure out what the code does, line-by-line, with "faith" that every

Given a C++ coded function fun(), e.g., mergesort(), fact(), etc.

University at Albany Computer Science Dept.
thing on \( N \) elements to be sorted.

Simply put, \( N \) is a number for which \texttt{mergesort()} from \texttt{DSO} does the right

\begin{verbatim}
[0..N-1]
data

post-conditions Mani and Savitch wrote for all cases of values in array

\texttt{DSO}’s \texttt{mergesort()} \texttt{int *data, int N } \texttt{WILL SATISFY the}

example of a property of a zero or positive integer \( N \):

arithmetic.

That “faith” is based on the principle of mathematical induction, an axiom of

University at Albany Computer Science Dept.
"This PROPERTY is true for ALL counting numbers.

THEN, people can conclude:

(b) IF the PROPERTY is true for all numbers less than "N" (a variable as in algebra), THEN it is verified for N itself.

AND

(a) It is TRUE for 0 and 1 (or the first few numbers including 0).

Suppose it is verified for a PROPERTY of counting numbers:
happens “no matter what is \( N \).”

EACH valid input involves a particular NUMBEr \( N \). The “right thing

matter what valid inputs are given, will do “the right thing”.

RELY on Mathematical Induction to conclude that ALL the computations, no

critical, of course

knowledge of C++ rules, careful reading of code, responsible memory use, etc is

deepER thought to verify that the code works.

USE the assumption (“faith”) that the recursive calls WORK (do no

the function you are studying.

(1) verify that the arguments to recursive calls involve FEWER elements than

(a) verify that the code works in the “Base Cases”.

Application to study recursive programs:

University at Albany Computer Science Dept.

6
Don't forget this... CSI310 uses the fact that the graph of the function $N \left( \frac{2}{1} \right) + \frac{N}{2} = 2 / (1 + N) N$

The sum of the first $N$ numbers 1, 2, \ldots, $N$

$N$ properties of $N$: The following equation is true:

$N$ typical application of Math. Induction in Math courses like CSI210:

University at Albany Computer Science Dept.
Combining is called "Merging".

4. Combine the two sorted groups into a large sorted list, and return it.

INDEPENDENTLY:
3. Sort each of these smaller groups (by recursive calls). That means size.

2. Divide the elements to be sorted into two groups of equal (or almost equal).

1. If the input list has length 0 or 1, return it ("base case").

paradigm or pattern is applied to the problem of sorting a sequence:

Mergesort is the algorithm that is invented when the divide-and-conquer
An array of 1 element IS SORTED (trivially)

BASE CASE: if \( n == 1 \) return 1;

\( \text{int } n, n2; \)

and local extent variables

Value parameters data and in mergeSort(char **data, int n)

In your project/DSO's code, the automatic variables were:

\( \text{is done} \):

very impoRTant::LOCAL EXTAnt (AUTOMATIC) variables are used to

Spring 2006 MergeSort Project: Double-Storage array-based variant.

University at Albany Computer Science Dept.
merge(data, nl, n2); MUST VERIFY SEPARATELY (it is not recursive).

merge_sort(data, nl, n2); WORKED?

merge_sort(data, nl); WORKED?

\[
\begin{align*}
0 & < u \\
\text{Since } n \leq 2, \quad & n < n \\
& < n \\
& > n \\
	herefore & \text{ Verify:} \\
& n \neq n - n1; \\
& n1 / n \neq n2.
\end{align*}
\]
Different activations—Different automatic

The storage used for automatic variables is in the activation record.

The event of "calling one function once" is an activation.

Review Way 2: Individual activations studied using a tree diagram to express interrelationships.

Each time a function is called, an activation record is created.

So, we do a careful analysis of the separate, individual recursive or non-recursive function activations.
4. Now, automatic variables are in the activation record of the activation

RETURNED TO.

3. The ACTIVATION whose CALL originally created this activation is

RETURNED TO.

2. This ACTIVATION'S ACTIVATION RECORD "goes away"

1. The return value (if any) is saved for use by the caller.

When a function ACTIVATION executes the RETURN operation: 

Only true "logically"... this data is actually stored in the called activation.

ACTIVATION is destroyed, and its Record gets recycled.

Really: When an ACTIVATION executes the return; operation, that spot within the function's body is this activation CALLED a function.

Different One whose call operation created this one. (3) The return.

Definition of Activation Record: The data structure that holds (1) An
will control what THAT activation does.

Really: A new Function Activation is created, and the function's body

Wrong: Control "jumps" or "goes to" the function's body.

What HAPPENS when the computer executes a FUNCTION CALL?
sorts a list of 7 letters.

Let's review the operation of the top level activation of Mergesort when it
The two recursions have FINISHED; the top level activation will now merge.

Their results:

- Input
Now, let's examine everything that happens during all the recursions.

Virtual language convention: The indicated computation or activation was FINISHED.

**BROWN**

\[
\begin{align*}
&\text{Recurse.} \\
&(G\ D\ A\ F\ C\ E\ B) \\
&(A\ B\ C\ D\ E\ F\ G) \\
&(A\ D\ F\ G) \\
&(B\ C\ E)
\end{align*}
\]

**MERGE**

\[
\begin{align*}
&(G\ D\ A\ F) \\
&\text{SPLIT} \\
&(G\ D\ A\ F\ C\ E\ B) \\
&(A\ B\ C\ D\ E\ F\ G)
\end{align*}
\]

\[
\begin{align*}
&(A\ D\ F\ G) \\
&\text{MERGE}
\end{align*}
\]
Input

SPLIT

Recurse.
(G D A F C E B)

(MERGE)

(ADFG)

(G D A F C E B)

(MERGE)
1. Plus the number of items you have to divide $N$ by two with rounding to get

$$\lceil \log_2(N) \rceil + 1$$

The number of levels is rounded $(\log_2(N) + 1)$. 

---

**Input size**

The total number of elements involved in each level is $N$ or fewer (the original $(N)$ operations is less than $(N)$ (number of elements) * constant). During any single activation, the time used to split, merge, and mix.

"All the work is done by MERGE."  

Why is MergeSort so Fast?

University at Albany Computer Science Dept.
When \( N = 2^{30} \) which is \( 10^8 \). \( \log(N) = 30 \).

where Constant is somewhat bigger constant then Constant.

\[ \text{Constant} \times \log(N) \]

So, the total time is LESS THAN
particular activation of the (recursive) function MERGESORT is doing.

VERY IMPORTANT: AUTOMATIC variables are used to keep track of what a
An expression tree illustrates the computations done when you evaluate an expression.

The merge sort algorithm uses the computations done by merge sort.
\[(6+9)/3 \times (6-4) = 15/3 \times 2 = 5 = 15/3\]

\[\left(\frac{(6+9)}{3}\right) \times (6-4) = 15/3 \times 2 = 5 = 15/3\]
\[(6+9) / 3 \times (6-4)] = 15 / 3 = 5\]
\[(6+9)/3*<6−4>\]

\[10=5*2\text{ DONE!}\]
During the run of a recursive function, the stack of activation records was to observe the stack to delete from a stack is called push, delete from a stack is called pop.

(Insert into a stack is called push, delete from a stack is called pop.

and non-recursive.

Other data relevant to all C/C++ function calls and returns, both recursive and non-recursive, other data relevant to all C/C++ programs.

Implementing and organizing local variables and

3. (The run-time stack of activation record, internal to the system when it runs C/C++ programs.

2. Storing and organizing intermediate results when evaluating expressions.

1. Figure which pairs of parentheses MATCH in a correctly nested parenthesis expression.

3 uses for stacks:

ONLY ONE END (called the top).

That access, insertion, and deletion are permitted at

What is a stack? A stack is a sequence that is restricted so

University at Albany Computer Science Dept.
(q) What is a tree?

(c) One arc is from this tree’s root to the root of each of the trees specified under (and) or the root.

(b) Zero or more rooted trees, with no nodes or arcs in common with each other.

(a) One root node (and) a rooted tree is a structure of nodes and arcs (pairs of nodes) that has:

What is a tree?
An expression: (q). are subexpressions of the expression.

(c) Any operator and operands under overlapping as operands or more expressions.

If it has an operator, it has one.

and

either is an identical or constant.

A tree has:

under (q).

the root of each of the trees specified

(c) One arc from this tree's root to

other or the root. (and)

no nodes or arcs in common with each

(q) Zero or more rooted trees, with

(a) One root node. (and)
XXX employees.

Sometimes with the complex C/C++ precedence/associativity rules, FIRE that obvious. (2) If a programmer you are supervising tries to show off his/her

2 Practical Rules: (1) If it's doubtful or subtle, USE PARENTHESES to make it

\[ \text{not } (3 \times 4 + 3) = 3 \times (4 \times 3 + 3) = 20 \]

precedence than addition. from elementary school:

must memorize or look up. They begin with "multiplication has higher

rather complicated OPERATOR precedence and associativity rules people

How do you know which operator is evaluated first?

\[ A = B + C \times D + E + F + G \]

Not fully parenthesized:

expressions.

To make learning these ideas easier, we will start with fully parenthesized

University at Albany Computer Science Dept.
The top level operation "Assgin to A" is executed last. Why must it be done last? It uses the results of the all previous operations.

1. Assign it also to A.
2. Assign the last sum to B.
3. Subtract that from C, remember result.
4. Multiply F and G.
5. Add subtraction's result to this last product.
6. Assign the last sum to B.

I. Increment E first.

means:

\[
A = B = C - D \times E + F \times G
\]

fully parenthesized:

\[
A = (B = C - (D \times (E + (F \times G))))
\]
Example of an expression and its Parse Tree

(A = (B = ((C - (D * (E ++ ))) + (F * G )))

Details cont’d (on the next 2 frames)

( = ) top level operator is assignment

(B = (C - (D * (E ++ ))) + (F * G )))

( = ) top level operator is assignment

(A = (B = (C - (D * (E ++ ))) + (F * G )))

(top level operator is mult.)

( * )

( + ) top level operator is addition

((C - (D * (E ++ ))) + (F * G )))

(top level operator is subtraction)

((C - (D * (E ++ ))) + (F * G )))

identifier B

identifier A

(identifier)

A

B

Example of an expression and its Parse Tree
top level operator is multiplication

\( F \times G \)
The top level operator in this expression is subtraction.

```
(C-(D*(E++)) )
```

- The operator `*` is the top level operator in the subexpression `(D*(E++)).
- The operator `++` is the top level operator in the subexpression `(E++).
- The operator `-` is the top level operator in the subexpression `(C-(D*(E++)))`.
(a) Any operator and operands under overlap(s) (and no operands as operands or more expressions) (b) If it has an operator, it has one singly (and or has a top level operator, exclude). Either is an identity or constant.

An expression under (a).

(c) One arc from this tree's root to the root of each of the trees specified (c) Other or the root (and) no nodes or arcs in common with each (b) Zero or more rooted trees, with (a) One root node (and) a tree has:

Definitions:

"Trees and expressions" Fit these to be "trees and expressions."

Your job: Check that these examples purporting
Definitions: "trees and expressions" fit these expressions to be "trees and expressions" fit these expressions. Your job: Check that these examples, purporting...
The operands are substitutional.

Clause (c) is OK.

Expression de: Clause (b) is OK?

Is I or more expressions as operands, not overlapping.

Top level operator is increment (++).

(E++)

Top level operator is multiplication (*).

(D*E++)

Top level operator is subtraction (-).

(C-D*(E++))

Has an operator:

Subtraction (c)
and return its result.

(3) Combine the results from (2) using the meaning of the operator to compute

(2) RECURSIVELY Call Evaluate($T_1$) for each of the trees

(1) If $T$ is just one node only, then the expression must be a constant or

Evaluate(ParseTree $T$)

The following recursive algorithm evaluates an expression when given its parse tree:

The “tree of an expression” is called the expression’s Parse Tree.

The (rather difficult and non-trivial) job of determining out the tree from a given

expression is called Parsing.

The tree of an expression represents the expression’s structure with

University at Albany Computer Science Dept.