Recurrent inequalities arise in the analysis of recursive algorithms. “Divide-and-conquer” is one of the most successful recursive techniques that leads to efficient algorithms.

In this note we consider a general technique to evaluate recurrent inequalities by “guessing” an upper bound. In the exercises below, $M(n)$ represents the cost of an algorithm on instances of size $n$, and $g(n)$ is a “guess function” which we use to bound $M(n)$ from above.

1. (Special case, arising from the Karatsuba-Ofman multiplication of integers.) Suppose the function $M(n) > 0$ satisfies the following recurrent inequality for $n \geq 2$:

\[ M(n) \leq 3M(\lfloor n/2 \rfloor) + Cn. \]  

(1)

Suppose moreover that the function $g(n)$ satisfies the reverse inequality (for $n \geq 2$):

\[ g(n) \geq 3g(\lfloor n/2 \rfloor) + Cn \]  

(2)

and also satisfies the initial condition

\[ g(1) \geq M(1). \]  

(3)

Prove by induction that for every $n$,

\[ M(n) \leq g(n). \]  

(4)

2. Use this result to prove: if $M(n)$ satisfies

\[ M(n) \leq 3M(\lfloor n/2 \rfloor) + O(n) \]  

(5)

then $M(n) = O(n^{\log 3})$. ($\log 3 = 1.5849.. < 1.585$)

3. (More general version - still not the “most general” form.) Suppose we have three functions $r(x), s(x), t(x)$ such that

(a) $r$ is monotone: if $x_1 \leq x_2$ then $r(x_1) \leq r(x_2)$;
(b) if \( x \geq 2 \) then \( s(x) < x \).

(There is no condition on \( t(x) \).)

Suppose now the function \( M(n) > 0 \) satisfies the following recurrent inequality for \( n \geq 2 \):

\[
M(n) \leq r(M(\lfloor s(n) \rfloor)) + t(n).
\]  

(6)

Suppose that the function \( g(n) \) satisfies the reverse inequality (for \( n \geq 2 \)):

\[
g(n) \geq r(g(\lfloor s(n) \rfloor)) + t(n)
\]

(7)

and also satisfies the initial condition

\[
g(1) \geq M(1).
\]

(8)

Prove by induction that for every \( n \),

\[
M(n) \leq g(n).
\]

(9)

(Note that the first result is a special case of the second: we need to set \( r(x) = 3x \), \( s(x) = x/2 \), \( t(x) = Cx \).)

4. Use this more general form to prove:

4.1 If

\[
M(n) \leq 7M(\lfloor n/2 \rfloor) + O(n^2)
\]

(10)

then \( M(n) = O(n^{\log 7}) \).  (\( \log 7 = 2.8073... < 2.81 \).)

(This function arises in the analysis of Strassen’s matrix multiplication algorithm.)

4.2 If

\[
M(n) \leq 2M(\lfloor n/2 \rfloor) + O(n)
\]

(11)

then \( M(n) = O(n \log n) \).

(This function arises in the analysis of MERGE-SORT.)

5. (A recurrent inequality arising in an \( O(n) \) algorithm to find the median.)

Suppose the function \( M(n) > 0 \) satisfies the following recurrent inequality:

\[
M(n) \leq M(\lfloor n/5 \rfloor) + M(\lfloor 7n/10 \rfloor) + O(n).
\]

(12)

Prove that \( M(n) = O(n) \).

Generalize the result of item 3 to fit this type of situation.