Massachusetts Institute of Technology — Handout 25
6.042J/18.062J: Mathematics for Computer Science — October 9, 1998
Professors Nancy Lynch and Srini Devadas

Notes for Recitation 5 (10/9/98)

1 Asymptotic Notations

In many parts of mathematics and computer science, one needs to compare growth rates of functions. There are about 6 ways to make such comparisons.

1.1 Basic asymptotic relations

The most common notations are $O()$, $\Theta()$, and $\Omega()$:

Definition 1.1 *For functions $f, g : R \to R$, define*

$$f = O(g) \text{ iff } \exists c > 0 \exists x_0 \forall x \geq x_0 : |f(x)| \leq cg(x).$$

Also

$$f = \Theta(g) \text{ iff } [f = O(g) \text{ and } g = O(f)],$$

and

$$f = \Omega(g) \text{ iff } g = O(f).$$

The value of these notations is that they highlight growth rate and allow suppression of distracting factors and low-order terms. For example, if the running time of an algorithm is

$$T(n) = 10n^3 - 20n^2 + 1,$$

then

$$T(n) = \Theta(n^3).$$

$T(n)$ grows *cubically*: it is bounded both above and below by functions of the form cn^3.

Another such example is

$$\pi^2 3^{x-7} + \frac{(2.7x^{113} + x^9 - 86)^4}{\sqrt{x}} - 1.08^{3x} = \Theta(3^x).$$

Lemma 1.2 *If $f = O(g)$, then $af = O(bg)$ for any $a, b > 0$*

Exercise: Prove Lemma 1.2.

If $l(x)$ is any function which increases unboundedly — $\ln x$ is a good example — then $x/l(x) < x$ as soon as x gets large enough that $l(x) > 1$. In particular, $x/l(x) = O(x)$. Intuitively, $f = O(g)$ means that, to within a constant factor, g grows at least as fast as f. In this case, the g grows strictly faster:

Theorem 1.3

It is not true that $x = O(x/\ln x)$.

Proof. We must show that for all $c > 0$, there does not exist an x_0 such that for all $x \geq x_0$, we have $|x| \leq cx/\ln x$. But $x > cx/\ln x$ for all $x > e^c$, so there can't be such an x_0. ∎

Notice that, again, the only fact we used about $\ln x$ in the above proof is that it is an unbounded increasing function.

1.2 Growth rates of common functions

A more significant fact about the growth of $\ln x$ is

Theorem 1.4

$$\forall a > 0 : \ln x = O(x^a).$$

Proof. Remember L'Hôpital's rule: if $\mathrm{Lim}_{x\to\infty} f(x) = \infty = \mathrm{Lim}_{x\to\infty} g(x)$, then

$$\mathrm{Lim}_{x\to\infty} f(x)/g(x) = \mathrm{Lim}_{x\to\infty} f'(x)/g'(x)$$

where f', g' are the derivatives of f, g, and f, g, f', g' are continuous. So

$$\mathrm{Lim}_{x\to\infty} \frac{\ln x}{x^a} = \mathrm{Lim}_{x\to\infty} \frac{1/x}{ax^{a-1}} = \mathrm{Lim}_{x\to\infty} \frac{1}{ax^a} = 0.$$

The last equality follow from the familiar fact – which we'll take for granted – that that for all $a > 0$, the value of x^a grows unboundedly as x approaches infinity. Since the limit is 0, we can be sure that once x is large enough, $\ln x/x^a \leq 1$, that is $\ln x \leq x^a$. So indeed $\ln x = O(x^a)$. ∎

Similarly,

Theorem 1.5

$$\forall a > 1 : x = O(a^x).$$

Proof. We know that $\ln a > 0$, so in the Taylor series

$$a^x = a^0 + x(\ln a)a^0 + x^2(\ln a)^2 a^0/2 + \ldots$$

all the coefficients are positive. Hence

$$a^x > 1 + (\ln a)x > (\ln a)x \text{ for } x > 0.$$

∎

Corollary 1.6

$$\forall k \in N, a > 1 : y^k = O(a^y).$$

Proof. By induction on k. In the base case $k = 0$, we have $y^k = y^0 = 1 < a \leq a^y$ for $y \geq 1$, so $y^0 = O(a^y)$. Notice that the only facts we have used here about a^y are that $a^1 = a$ and a^y is a nondecreasing function of y.

We now assume the induction hypothesis $y^k = O(b^y)$ for all $b > 1$ and prove $y^{k+1} = O(a^y)$ for all $a > 1$. For any $a > 1$, let $b = \sqrt{a}$, so also $b > 1$. By hypothesis, $y^k \leq cb^y$ for some $c > 0$ and all $y \geq$ some y_0. Also, by Theorem 1.5, $y \leq c'b^y$ for some $c' > 0$ and all $y \geq$ some y'_0. So

$$\begin{aligned} y^{k+1} &= y(y^k) \leq c'b^y(cb^y) = cc'(b^y)^2 \\ &= cc'b^{2y} = cc'(b^2)^y = cc'a^y \end{aligned}$$

for $y \geq \max(y_0, y'_0)$, which shows that $y^{k+1} = O(a^y)$. ■

1.3 A note on O notation

Often one sees expressions like:

$$(1 + 1/x)^n = 1 + n/x + O(1/x^2).$$

The above expression really means that

$$(1 + 1/x)^n - 1 - n/x = O(1/x^2).$$

That is, the error term in the approximation for $(1 + 1/x)^n$ is at most a constant divided by x^2.

1.4 Induction and asymptotic notation

One has to be careful when combining inductive proofs with asymptotic notation. Here is an example of a wrong proof. Recall that F_n denotes the n-th Fibonacci number.

False Theorem 1 $F_n = O(1)$

Proof. The proof is by strong induction. Let $P(n)$ be the predicate that $F_n = O(1)$. In the base case, $P(1)$ is true because $F_1 = 1 = O(1)$. In the inductive step, for $n \geq 1$ assume that $F_1 = O(1), F_2 = O(1), \ldots, F_n = O(1)$ to prove that $F_{n+1} = O(1)$. There are two cases. If $n = 1$, then $F_{n+1} = F_2 = 1 = O(1)$. If $n > 1$, then $F_{n+1} = F_n + F_{n-1} = O(1) + O(1) = O(1)$. ■

Where is the bug?! The proof is actually correct; the problem is that it is not a proof of the theorem! What we actually proved is that $\forall n : F_n = O(1)$. It is in fact the case that $\forall n : P(n)$ i.e. $\forall n : F_n = O(1)$ is true. This is because once we fix n, i.e. within the quantifier, F_n is just a specific number such as F_{100} or F_{1000} etc. so it is a constant and hence we can say it is $O(1)$. However, this does not imply the same about the *function* $f(n) = F_n$ which is what we care about.

1.5 Other asymptotic relations

There are three other asymptotic notations in common use. We include them here for the record, but will not use them much this term.

The tightest of the relationships between growth rates is *asymptotic equality*, $\sim$:

Definition 1.7

$$f \sim g \ \textit{iff} \ Lim_{x\to\infty} \frac{f(x)}{g(x)} = 1.$$

This was the relation described in lecture for the Harmonic number

$$H_n \sim \ln n.$$

For example, $x + 1/x \sim x - \sqrt{x}$. Note that if the values of f and g are nonegative, then $f \sim g \implies f = \Theta(g)$.

Exercise: Give an example of f, g such that $f = \Theta(g)$ but **not** $f \sim g$.

To indicate that g actually grows even faster than f, we can use "Little Oh"notation:

Definition 1.8 $f = o(g)$ *iff* $\forall c > 0 \exists x_0 \forall x \geq x_0 : |f(x)| \leq cg(x)$.

This definition is a little hard to follow because of all the quantifiers. For functions, g, which are nonnegative — which is generally all we care about in this context — a clearer way to say $f = o(g)$ is:

$$\mathrm{Lim}_{x\to\infty} \frac{f(x)}{g(x)} = 0.$$

For example, the proof of Theorem 1.3 above actually yielded the stronger result:

Theorem 1.9

$$\frac{x}{\ln x} = o(x).$$

Theorem 1.10

$$\textit{It is not true that } \frac{x}{100} = o(x).$$

Proof. The proof is by contradiction. Assume for the purpose of contradiction that $x/100 = o(x)$. Then for all $c > 0$, there exists an x_0 such that for all $x \geq x_0$, $|x/100| \leq cx$. In particular, if $c = 1/200$, there must exist an x_0 such that for all $x \geq x_0$, $|x/100| \leq x/200$. Since this statement is false for all positive x, we have reached a contradiction. ∎

Another, rarely used, asymptotic notation is "Little Omega", $\omega()$, which is just the opposite of "Little Oh", $o()$.

Definition 1.11 $g = \omega(f)$ *iff* $f = o(g)$

For example, $x^2 = \omega(x)$ since x^2 grows strictly faster than x.

2 Some Sums

2.1 Perturbation method

We have seen some methods to find closed forms of sums. Finding closed form solutions is tricky business because it is not always the case that they exist! Let's carefully review the way to discover summation formulas called the *perturbation* method in lecture. As with all other methods it only works sometimes.

Consider the sum, $S_n = \sum_{i=0}^{n} ix^i$.

The general approach is to add the $(n+1)$st term to both sides,

$$\begin{aligned} S_n + (n+1)x^{n+1} &= 0x^0 + \textstyle\sum_{i=1}^{n+1} ix^i \\ &= \textstyle\sum_{i=0}^{n}(i+1)x^{i+1} \\ &= \textstyle\sum_{i=0}^{n} ix^{i+1} + \sum_{i=0}^{n} x^{i+1} \\ &= x\textstyle\sum_{i=0}^{n} ix^i + x\sum_{i=0}^{n} x^i \\ &= xS_n + x(1-x^{n+1})/(1-x) \end{aligned}$$

Having found S_n in the last expression, bring terms involving S_n to one side and solve,

$$(1-x)S_n = \frac{x(1-x^{n+1}) - (n+1)x^{n+1}(1-x)}{(1-x)}$$

$$S_n = \frac{x - (n+1)x^{n+1} + nx^{n+2}}{(1-x)^2}$$

2.2 Integral method

Let us now go back to the integral method and see how to evaluate the sum

$$\sum_{i=1}^{\infty} \frac{1}{i^2}$$

In a similar fashion to the example in the lecture, this sum, minus its first term, is less than the area under the curve $y = 1/x^2$ from 1 to ∞. That is,

$$\begin{aligned} \sum_{i=1}^{\infty} \frac{1}{i^2} &= 1 + \sum_{i=2}^{\infty} \frac{1}{i^2} \\ &\leq 1 + \int_1^{\infty} x^{-2} dx \\ &= 2. \end{aligned}$$

Thus, our infinite sum is at most 2, and in particular it converges. In fact, one can prove (using complex analysis) that the exact value of the sum is $\pi^2/6$; we won't prove this here.

2.3 Formal manipulation

Next let's look at the following *doubly infinite* sum

$$\sum_{i=0}^{\infty}\sum_{j=0}^{\infty} x^i y^j,$$

where (critically)

$$0 < x, y < 1.$$

We can therefore re-write this as

$$\begin{aligned}\sum_{i=0}^{\infty}\sum_{j=0}^{\infty} x^i y^j &= \sum_{i=0}^{\infty} x^i \left(\sum_{j=0}^{\infty} y^j\right)\\ &= \sum_{i=0}^{\infty} x^i \left(\frac{1}{1-y}\right)\\ &= \left(\frac{1}{1-y}\right)\sum_{i=0}^{\infty} x^i\\ &= \frac{1}{(1-x)(1-y)}.\end{aligned}$$

2.4 And some you can't sum...

We note that the manipulations above were justified because we had a series of positive real numbers whose sum converged. Just to reinforce this, here's a case in which re-arrangements of the sum are clearly not allowed. Let

$$S = \sum_{i=0}^{\infty}(-1)^i = 1 + (-1) + 1 + (-1) + \cdots .$$

Consider the following:

$$S = [1 + (-1)] + [1 + (-1)] + \cdots = 0 + 0 + \cdots = 0.$$

$$S = 1 + [(-1) + 1] + [(-1) + 1] + \cdots = 1 + 0 + 0 + \cdots = 1.$$

What's the problem here?

If we define $S_n = \sum_{i=0}^{n}(-1)^i$ then $S_n = 1$ for odd n and $S_n = 0$ for even n. Hence S_n does not have a well-defined limit as n tends to infinity. S is not a convergent sum.

3 Akra-Bazzi and the Master Theorem

3.1 Applying Akra-Bazzi

One difficulty in applying Akra-Bazzi is the problem of finding p. This is hard because you have to solve the hard equation $\Sigma_{i=1}^{k} a_i b_i^p = 1$. You might approach this problem by guessing a value for p and computing the sum. If the result is greater than 1, then try a larger value of p; if the result is smaller than 1, try a smaller value. However, you might be unlucky and never find the exact value.

Sometimes, computing p is unnecessary. Suppose that $g(x) = x^q$, where $q > p$. Then $T(x)$ can be computed as follows:

$$\begin{aligned} T(x) &= \Theta(x^p(1 + \textstyle\int_1^x \frac{u^q}{u^{p+1}} du)) \\ &= \Theta(x^p(1 + \textstyle\int_1^x u^{q-p-1} du)) \\ &= \Theta(x^p(1 + \tfrac{1}{q-p}(x^{q-p} - 1))). \end{aligned}$$

Since $q > p$, the term x^{q-p} dominates, so we get:

$$T(x) = \Theta(x^p x^{q-p}) = \Theta(x^q).$$

The solution does not depend on the hard-to-compute value p. Thus, if $g(x) = x^q$, a good trick is to compute $\Sigma_{i=1}^{k} a_i b_i^q$. If the result is less than 1, then we know $q > p$ and so the solution is $\Theta(x^q)$.

For instance, for the recurrence $T(x) = T(x/3) + T(x/4) + x$, we have $g(x) = x^1$, so $q = 1$. Since $(1/3)^1 + (1/4)^1 < 1$, we know that $p < q$ and the solution is $\Theta(x^q)$.

3.2 The Master Theorem

The "Master Theorem" is commonly used in discrete math and algorithms courses. See the presentation in the Cormen, Leiserson, Rivest book for 6.046. It can be used to solve many recurrences of the form:

$$T(n) = aT(\lceil n/b \rceil) + f(n)$$

where $T, f : N \to N$, $a > 0$, and $b > 1$. Namely, let $\beta = \log_b a$. Then

$$T(n) = \begin{cases} \Theta(n^\beta) & \text{if } f(n) = O(n^{\beta-\epsilon}) \text{ for some } \epsilon > 0, \\ \Theta(n^\beta \log n) & \text{if } f(n) = \Theta(n^\beta), \\ \Theta(f(n)) & \text{if } af(n/b) \le cf(n) \text{ for some } c < 1 \text{ and all large } n. \end{cases}$$

The third condition looks different, but it does not overlap with the others, because if $af(n/b) \le cf(n)$ for some $c < 1$ and all large n, then

$$f = \Omega(n^{\beta+\epsilon})$$

for $\epsilon = -\log_b c$; note that $\epsilon > 0$ because $c < 1$. (We leave the proof of this claim as an exercise.) The converse, however, does not hold, so we can't replace the stronger condition used in the Master Theorem by this last condition.

Solution to the Exercise:

Assume for simplicity that n is a power of b, say $n = b^k$, and the "smoothness" condition $af(n/b) \leq cf(n)$ holds for all n.

So

$$\begin{aligned} f(n) &\geq (a/c)\, f(n/b) \geq (a/c)\,((a/c)f((n/b)/b)) = (a/c)^2 f(n/b^2) \\ &= \cdots = (a/c)^k f(n/b^k) = (a/c)^k f(1). \end{aligned}$$

But

$$\begin{aligned} (a/c)^k &= (b^{\log_b a - \log_b c})^k = (b^k)^{(\log_b a - \log_b c)} \\ &= n^{\log_b a - \log_b c}, \end{aligned}$$

so

$$f(n) = \Omega((a/c)^k) = \Omega\left(n^{\log_b a - \log_b c}\right) = \Omega\left(n^{\log_b a + \epsilon}\right)$$

where $\epsilon = -\log_b c$.

The Master Theorem applies nicely to the common case when $f(n) = \Theta(n^\alpha)$.

Corollary 3.1 *If $T(n) = aT(\lceil n/b \rceil) + f(n)$ where $a > 0, b > 1$ and $f(n) = \Theta(n^\alpha)$, then*

$$T(n) = \begin{cases} \Theta(n^\beta) & \text{if } \alpha < \beta, \\ \Theta(n^\beta \log n) & \text{if } \beta = \alpha, \\ \Theta(n^\alpha) & \text{if } \alpha > \beta. \end{cases}$$

Proof.

Case 1: $\beta > \alpha$.

Choose $\epsilon = \beta - \alpha > 0$. We have that $\alpha = \beta - \epsilon$; hence $f(n) = \Theta(n^\alpha) = \Theta(n^{\beta - \epsilon})$ with $\epsilon > 0$. So the first case of the Master Theorem applies, yielding

$$T(n) = \Theta(n^\beta),$$

as desired.

Case 2: $\beta = \alpha$.

Then $f(n) = \Theta(n^\beta)$, so the second case of the Master Theorem applies, yielding

$$T(n) = \Theta(n^\beta \log n),$$

as desired.

Case 3: $\alpha > \beta$.

Let's prove this case first when $f(n) = dn^\alpha$ for some $d > 0$ and all large n.

Let $c = a/b^\alpha = b^\beta/b^\alpha = 1/b^{\alpha-\beta}$. We know $b > 1$ and $\alpha - \beta > 0$, so $b^{\alpha-\beta} > 1$. Therefore $c = 1/b^{\alpha-\beta} < 1$. Also,

$$af(n/b) = ad(n/b)^\alpha = (a/b^\alpha)dn^\alpha = cf(n).$$

So the third case of the Master Theorem applies, yielding

$$T(n) = \Theta(f(n)) = \Theta(n^\alpha),$$

as desired.

Now suppose all we know is that $f(n) = \Theta(n^\alpha)$. Then by definition of $\Theta()$, there are constants $n_0, d_1, d_2 > 0$ such that $d_1 n^\alpha \leq f(n) \leq d_2 n^\alpha$ for all $n > n_0$. For $i = 1, 2$, let $f_i(n) = f(n)$ for $n \leq n_0$, and for $n > n_0$, let $f_i(n) = d_i n^\alpha$. So $f_1(n) \leq f(n) \leq f_2(n)$ for *all* $n \in N$, and also $f_i(n) = d_i n^\alpha$ for all $n > n_0$.

Now define T_1, T_2 by the conditions: $T_i(0) = T(0), T_i(1) = T(1)$, and $T_i(n) = aT_i(\lceil n/b \rceil) + f_i(n)$ for $n > 1$. Since $f_i(n)$ equals a constant times n^α for all large n, our earlier reasoning shows that

$$T_i(n) = \Theta(n^\alpha)$$

for $i = 1, 2$. But since $f_1(n) \leq f(n) \leq f_2(n)$ for all n, it follows (by an easy induction) that

$$T_1(n) \leq T(n) \leq T_2(n),$$

which immediately implies that $T(n) = \Theta(n^\alpha)$.

■

It's helpful to note that the Master Theorem and its corollary still hold if, instead of rounding n/b up to $\lceil n/b \rceil$, we rounded down to $\lfloor n/b \rfloor$, or even chose to round up or down as a function of n.

3.3 Master theorem example

Strassen, in 1972, discovered a way of multiplying two $n \times n$ matrices by performing only seven multiplications of linear combinations of their eight $n/2 \times n/2$ submatrices and then computing eight linear combinations of the seven product matrices. Since a linear combination of a fixed number of $n/2 \times n/2$ matrices requires $O(n^2)$ arithmetic operations, the number, $T_3(n)$, of arithmetic operations required by Strassen's procedure is

$$T_3(n) = 7T_3(\lceil n/2 \rceil) + O(n^2).$$

The first case of the Master Theorem applies with $\log_b a = \log_2 7 > 2$, so

$$T_3(n) = \Theta(n^{\log_2 7}) = O(n^{2.81}).$$

This is a considerable improvement over the standard row-by-column method of matrix multiplication, which takes $\Theta(n^3)$ operations.

Strassen's technique has been extended in various *ad hoc* ways, now it is actually known how to multiply $n \times n$ matrices with $O(n^{2.49})$ operations. However, no one believes the currently fastest method is optimal. It remains an open question whether $n \times n$ matrices can be multiplied in $\Theta(n^2)$ operations.

3.4 Special case of the Master Theorem derived from Akra-Bazzi

We show how an important special case of the Master Theorem can be derived from Akra-Bazzi:

$$T(n) = aT(n/b) + n^{\alpha}$$

where $a > 0$ and $b > 1$. First we find p such that $a(\frac{1}{b})^p = 1$. In this case, $a = b^p$ or $p = \log_b a$.

Then from the Akra-Bazzi theorem,

$$T(n) = \theta(n^p(1 + \int_1^n \frac{u^{\alpha}}{u^{p+1}} du))$$

$$= \theta(n^p(1 + \int_1^n u^{\alpha-p-1} du))$$

In the case when $\alpha \neq p$,

$$= \theta(n^p(1 + \left[\frac{u^{\alpha-p}}{\alpha - p}\right]_1^n))$$

$$= \theta(n^p(1 + \frac{1}{\alpha - p}(n^{\alpha-p} - 1))) = \theta((1 - \frac{1}{\alpha - p})n^p + \frac{1}{\alpha - p}n^{\alpha})$$

Assuming a, b, α are fixed constants (i.e., don't change with n),

$$T(n) = \theta(n^p + n^{\alpha}) = \theta(n^{\log_b a} + n^{\alpha})$$

If $\alpha = p$, then

$$T(n) = \theta(n^p(1 + \int_1^n \frac{1}{u} du))$$

$$= \theta(n^p(1 + \ln n)) = \theta(n^{\alpha} \log n)$$

So all together we have,

$$T(n) = \begin{cases} \theta(n^{\log_b a}) & \text{if } \log_b a > \alpha \\ \theta(n^{\alpha} \log n) & \text{if } \log_b a = \alpha \\ \theta(n^{\alpha}) & \text{if } \alpha > \log_b a \end{cases}$$

For some intuition on the Master Theorem, see Cormen, Leiserson, Rivest pages 65-67. They break down $T(n)$ into some costs associated with nodes in a tree. In equation (4.6), then give two expressions, one for the total costs associated with the leaves and one for the total costs associated with the internal nodes. Depending on the relative values of a and b, one or the other of these expressions can dominate, or then can give equal contributions.

4 Team Problems

Problem 1. We begin with two large glasses. The first glass contains a pint of water, and the second contains a pint of wine. We pour 1/3 of a pint from the first glass into the second, stir up the wine/water mixture in the second glass, and then pour 1/3 of a pint of the mix back into the first glass. We now repeat this pouring back-and-forth process a total of n times.

(a) Describe a closed form formula for the amount of wine in the first glass after n back-and-forth pourings.

(b) What is the limit of the amount of wine in each glass as n approaches infinity?

Solution: The state of the system of glasses/wine/water at the beginning of a round of pouring and pouring back is determined by the total amount of wine in the first glass. Initially, the first glass contains one pint of water and no wine. Suppose the first glass contains w pints of wine, $0 \leq w \leq 1$. Since we're at the beginning of a round, it contains one pint of liquid, and therefore contains $1 - w$ pints of water. The second glass contains the rest of the wine and water.

Pouring 1/3 pint from first glass to second leaves 2/3 pint of liquid and $(2/3)w$ wine in the first glass, and 4/3 pints liquid and $1 - (2/3)w$ wine in the second glass. Pouring 1/3 pint back from second into first transfers a proportion of $(1/3)/(4/3)$ of the wine in the second glass into the first. So the round completes with both glasses containing a pint of liquid, and the first glass containing

$$(2/3)w + (1/4)(1 - (2/3)w) = 1/4 + w/2$$

pints of wine. After the second round, the first glass contains

$$1/4 + (1/4 + w/2)/2 = 1/4 + 1/8 + w/2^2$$

pints of wine, and after the nth round

$$\begin{aligned}
w/2^n + \Sigma_{i=1}^{n}(1/2)^{i+1} &= w/2^n + (1/2)\Sigma_{i=1}^{n}(1/2)^i \\
&= w/2^n + (1/2)(-1 + \Sigma_{i=0}^{n}(1/2)^i) \\
&= w/2^n + (1/2)(-1 + (1 - (1/2)^{n+1})/(1 - 1/2)) \\
&= w/2^n - 1/2 + 1 - (1/2)^{n+1} \\
&= w/2^n + 1/2 - (1/2)^{n+1}.
\end{aligned}$$

So when $w = 0$ initially, the pints of wine in the first glass is

$$1/2 - (1/2)^{n+1}.$$

You would expect after a thorough mixing that the glasses would contain the same amount of wine, viz. 1/2 pint. Indeed, the limiting amount of wine in the first glass approaches 1/2 from below as n approaches infinity. In fact, it approaches 1/2 no matter how the wine was initially distributed.

Problem 2. **(a)** Give an example of functions $f, g : N \to N$ such that neither $f = O(g)$ nor $g = O(f)$. Prove it.

Solution: $f(n) = 1 + n(n \bmod 2), g(n) = 1 + n((1 + n) \bmod 2)$. So $f(n)$ is n times the size of $g(n)$ on odd n, vice-versa on even n.

(b) Prove or give a counterexample: If $f_1, f_2, \ldots$ is a sequence of functions from N to N such that $f_i(n) = O(1)$ for all $i \geq 1$, then

$$\sum_{i=1}^{n} f_i(n) = O(2^n).$$

Solution: False, let f_i be a constant function with $f(n) = b(i)$ for all n, where $b(i)$ is nonnegative and not $O(2^i)$, for example, $b(i) = i2^i$. So $f_i = O(1)$ for each i. Now

$$\sum_{i=1}^{n} f_i(n) \geq f_n(n) = b(n) \neq O(2^n).$$

Problem 3. **(a)** Show that the Master Theorem does not apply to

$$S(m) = S(m/2) + (\log m)^2.$$

(b) Find a $\Theta()$ bound for S anyway.

(c) Find a $\Theta()$ bound for

$$T(n) = T(\sqrt{n}) + (\lg \lg n)^2.$$

The base condition is $T(2) = 1$.

Solution:

(a) Use $a = 1, b = 2$, so $\beta = 0$, and $n^\beta = 1$. The first two cases of master theorem obviously don't apply since $f(n)$ grows much faster than $\Theta(1)$. Now let's consider the third case. (assume $\log m$ is base 2)

$$\begin{array}{lcl} af(\frac{m}{b}) & \leq & cf(m) \\ f(\frac{m}{2}) & \leq & cf(m) \\ (\log \frac{m}{2})^2 & \leq & c(\log m)^2 \\ (\log m - 1)^2 & \leq & c(\log m)^2 \\ (\frac{\log m - 1}{\log m})^2 & \leq & c \end{array}$$

For large n, $\frac{\log m-1}{\log m}$ becomes 1, so c must be greater than or equal to 1. Therefore, we can not use the third case of master theorem since $c < 1$ does not exist.

(b) Assume $\log m$ is base 2.

$$\begin{aligned} S(m) &= S(\frac{m}{2}) + (\log m)^2 \\ &= (\log m)^2 + (\log \frac{m}{2})^2 + (\log \frac{m}{4})^2 \cdots (\log 2)^2 + S(1) \\ &= \sum_{i=0}^{(\log m)-1} (\log \frac{m}{2^i})^2 + S(1) \\ &= \sum_{i=0}^{(\log m)-1} (\log m - i)^2 + S(1) \\ &= \sum_{i=1}^{\log m} i^2 + S(1) \\ &= \frac{1}{6} \log m(\log m + 1)(2 \log m + 1) + 1 \end{aligned}$$

So, $S(m)$ is $\Theta((\log m)^3)$.

This also works with Akra-Bazzi.

(c) Substitute $n = 2^m$ and note that $m = \log n$.

$$T(2^m) = T(2^{\frac{m}{2}}) + (\log m)^2.$$

So let $S(m) = T(2^m)$. Now,

$$\begin{aligned} T(n) = T(2^m) = S(m) &= S(\frac{m}{2}) + (\log m)^2 \\ &= \Theta((\log m)^3) \\ &= \Theta((\log\log n)^3) \end{aligned}$$

So, $T(n)$ is $\Theta((\log\log n)^3)$.