Math 192r, Problem Set #9 (solutions)

1. *For various small values of n (1 through 5, at least), determine the average number of times that a $2n$-step Dyck path returns to the line $y = 0$ (counting $(2n, 0)$ as a return but not $(0, 0)$), and conjecture a general formula. Compute at least one more value to confirm (or disprove) your conjecture. (I strongly encourage you to do this problem symbolically, if possible, using a system like Maple or Mathematica to generate and study a list whose elements are representations of the Dyck paths. Take advantage of the recursive construction of Dyck paths discussed in class, in combination with the list-manipulation operations that are available to you in these computer algebra systems.)*

   Next, I'll show how you can explore the problem algebraically, without actually representing the Dyck paths combinatorially. Let us create a two-variable generating function in which a $2n$-step Dyck path that returns to the horizontal axis exactly k times is given weight $x^n y^k$. Its generating function is $1 + yP(x) + y^2[P(x)]^2 + y^3[P(x)]^3 + \ldots = 1/(1-yP(x))$, where $P(x)$ is the generating function for primitive Dyck paths: $P(x) = xS(x) = (1 - \sqrt{1-4x})/2$. So the generating function we want is

   $$R(x, y) = \frac{1}{1 - y\frac{1-\sqrt{1-4x}}{2}}.$$

   If we differentiate this with respect to y, and then put $y = 1$, we'll obtain a generating function in which the coefficient of x^n is the sum, over all Dyck paths of length n, of the number of returns to the horizontal axis. We have

   $$R_2'(x, 1) = 2\,\frac{1 - \sqrt{1-4x}}{(1 + \sqrt{1-4x})^2}.$$

   Using Maple, we write

```
Q := 1/(1-y*(1-sqrt(1-4*x))/2);
R := subs(y=1,Q);
S := subs(y=1,diff(Q,y));
Rt := taylor(R,x,10);
St := taylor(S,x,10);
seq(coeff(St,x,n)/coeff(Rt,x,n),n=0..9);
```

and obtain the sequence of fractions 0, 1, 3/2, 9/5, 2, 15/7, 9/4, 7/3, 12/5, 27/11. Multiplying by 2,3,4,... respectively, we get 0, 3, 6, 9, 12, 15, 18, 21, 24, 27, which we recognize. So we conjecture that the coefficient of x^n in `St`, divided by the coefficient of x^n in `Rt`, equals $3n/(n+2)$. That is, the average number of returns to the horizontal axis is exactly $3n/(n+2)$.

Before I show you how to use Maple to prove this, let's mimic everything we've done combinatorially. We'll represent each Dyck path by a list of 1's and -1's, and we'll represent the set of all such paths by a list of such lists. Thus we'll have the data-structures

```
[[]]
[[1,-1]]
[[1,-1,1,-1],[1,1,-1,-1]]
```

representing the set of all 0-step, 2-step, and 4-step Dyck paths, respectively. A Dyck path of size n (that is, with $2n$ steps) is given by a primitive Dyck path of size k followed by an unconstrained Dyck path of size $n-k$, with k ranging from 1 to n inclusive. Moreover, all such Dyck paths that arise in this way are distinct (we haven't over-counted any of them). A primitive Dyck path of size n (with $n > 1$) is just a $+1$ followed by an unconstained Dyck path of size $n-1$ followed by a -1. So we may define

```
All := proc(n) local k, answer;
    if n=0 then answer:=[[]];
    else
    answer := [];
    for k from 1 to n do
    answer := [op(answer),op(Combine(Prim(k),All(n-k)))];
    od; fi; RETURN(answer); end;
Prim := proc(n)
    Combine(Combine([[+1]],All(n-1)),[[-1]]); end;
Combine := proc(a,b) local i,j;
    [seq(seq([op(a[i]),op(b[j])],j=1..nops(b)),i=1..nops(a))]
    end;
```

Now we could write a routine that counts the number of returns to the horizontal axis:

```
Returns := proc(path) local i, partialsum, count;
    count := 0;
    partialsum := 0;
    for i from 1 to nops(path) do
        partialsum := partialsum + path[i];
        if partialsum = 0 then count := count + 1; fi;
    od; RETURN(count); end;
```

(And one nice feature of Maple is that we can test all these components out on real data, without having to create test-harnesses.) Lastly, we write a routine that, given a list of Dyck paths, computes the average number of returns to the horizontal axis:

```
Average := proc(a) local i;
    add(Returns(a[i]),i=1..nops(a))/nops(a); end;
```

(Note the use of `add` instead of `sum`; the latter should be used only in situations where you want Maple to use its smarts to simplify a sum that can be expressed in closed form.) Typing

```
seq(Average(All(i)),i=0..9);
```

gives the same sequence of fractions as before (though it takes the computer a bit longer).

Now let's see how Maple can help us prove the theorem. We have two generating functions, $A(x)$ and $B(x)$, where

$$A(x) = \frac{1 - \sqrt{1-4x}}{2x}$$

and

$$B(x) = 2\,\frac{1 - \sqrt{1-4x}}{(1 + \sqrt{1-4x})^2}.$$

We want to show that their respective coefficients satisfy $b(n) = \frac{3n}{n+2}a(n)$. That is, we want to prove

$$(n+2)b(n) = (3n)a(n).$$

Let's prove this by turning both sides into generating functions. If we differentiate $B(x) = \sum_{n\geq 0} b(n)x^n$ we get $\sum_{n\geq 0} nb(n)x^{n-1}$, so $\sum_{n\geq 0}(n+2)b(n)x^n = xB'(x) + 2B(x)$. Likewise $\sum_{n\geq 0}(3n)a(n)x^n = 3xA'(x)$. So we'll be done if we can demonstrate that $(x+2)B(x) = 3xA(x)$. How can Maple help us prove this? One thing we definitely shouldn't do is define $A(x)$ and $B(x)$ as functions of x! Maple knows how to simply algebraic expressions, not programs or functions. Even if we define things as expressions, we can still go astray. For instance, it might seem sensible to proceed like this:

```
A := (1-sqrt(1-4*x))/(2*x);
B := 2*(1-sqrt(1-4*x))/(1+sqrt(1-4*x))^2;
evalb(x*diff(B,x)+2*B=3*x*diff(A,x));
```

Maple returns the answer `false`, because it doesn't see a reason why the two should be equal. We haven't asked Maple to work hard at simplifying either side, so why should it? (I still haven't figured out which simplifications Maple does without one's asking; my version of Maple says `x+y=y+x` is true but `(x-y)*(x+y)=x*x-y*y` is "false".)

A better question to ask Maple (after defining `A` and `B`) is

```
evalb(simplify(x*diff(B,x)+2*B)=simplify(3*x*diff(A,x)))
```

But Maple still says this is false (because Maple has no standard form for algebraic functions of x, and so it simplifies the two sides of the equation in two different ways).

The best way to get Maple to go all-out and try to prove that two things are equal is to ask it to simplify the difference (to see if it gets 0) or to simplify the quotient (to see if it gets 1). But even that sometimes fails! Maple is a partner in doing algebra, not an infallible genii. In this case, `simplify((x*diff(B,x)+2*B)-(3*x*diff(A,x)))` gives 0 but `simplify((x*diff(B,x)+2*B)/(3*x*diff(A,x)))` doesn't give 1. (But

```
expand(1/(simplify((x*diff(B,x)+2*B)/(3*x*diff(A,x)))));
```

does give 1!)

2. *Let T_n be the number of domino tilings of a 3-by-2n cylinder, obtained by gluing together the left and right sides (of length 3) of a 3-by-2n rectangle.*

   (a) *Find a generating function for the sequence T_1,T_2,T_3,... .*

   It is easy to see that if a vertical line cuts through an even number of horizontal dominos, the adjacent vertical lines cut through an odd number of horizontal dominos, and vice versa. So there are two classes of tilings of the cylinder: those in which the 1st, 3nd, 5th, ... cuts pass through an odd number of horizontal dominos and the 2nd, 4th, 6th, ... cuts pass through an even number of horizontal dominos, and those in which the reverse occurs. Moreover, these two classes of tilings have the same cardinality (if we shift a tiling of the first type one step around the cylinder, we get a tiling of the second type, and vice versa). So we will be able to simplify our work if we calculate the number of tilings of one type and then double at the end to obtain the total number of tilings.

   Use the T, B, L, R coding for domino tilings. Let's assume that we're dealing with the class of tilings in which the leftmost column of symbols has an even number of dominos sticking out to the left and an odd number of dominos sticking out to the right. The allowed column-symbols are LLL, TBL, LTB, RRL, LRR, and RLR. (Don't get confused here: a square marked L is a square that is the Left half of a horizontal domino, and that domino is going to stick out to the *right*.) Two columns to the right of this column, these are also the allowed symbols. Here is the transition matrix:

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

If we define $Q(t) = \det(I - tQ)$, we get $Q(t) = (1-t)^2(1-4t+t^2)$ and

$$\frac{-2tQ'(t)}{Q(t)} = \frac{4t}{1-t} + \frac{8t-4t^2}{1-4t+t^2}$$

(here we've included an extra factor of 2, as discussed in the preceding paragraph).

(b) *Since the sequence satisfies a linear recurrence, there is a natural way to run the recurrence backward, obtaining values for T_0, T_{-1}, T_{-2}, etc. Compute T_n for all n between -10 and $+10$.*

First we'll compute T_n for all n between 1 and 4.

```
taylor(2*t/(1-t)+(4*t-2*t^2)/(1-4*t+t^2),t,5);
```

gives us a Taylor expansion whose coefficients are the numbers 6, 16, 54, 196. The sequence $T_1, T_2, T_3, \ldots$ is annihilated by the operator $I - 6T + 10T^2 - 6T^3 + T^4$ (that's what you get when you multiply the denominators $1-t$ and $1-4t+t^2$ and replace the variable t by the shift-operator T; sorry about the proliferation of T's in this problem!). We can embody the forward recursion $T_n = 6T_{n-1} - 10T_{n-2} + 6T_{n-3} - T_{n-4}$ and the backward recursion $T_n = 6T_{n+1} - 10T_{n+2} + 6T_{n+3} - T_{n+4}$ in Maple code:

```
T := proc(n)
    if n=1 then 6;
    elif n=2 then 16;
    elif n=3 then 54;
    elif n=4 then 196;
    elif n>4 then 6*T(n-1)-10*T(n-2)+6*T(n-3)-T(n-4);
    else 6*T(n+1)-10*T(n+2)+6*T(n+3)-T(n+4);
    fi; end;
```

Then, typing `seq(T(n),n=-10..10)` we get the list 524176, 140454, 37636, 10086, 2704, 726, 196, 54, 16, 6, 4, 6, 16, 54, 196, 726, 2704, 10086, 37636, 140454, 524176.

(c) *Formulate a conjecture based on your data.*

It is natural to conjecture that $T(-n) = T(n)$.

(d) *Prove your conjecture.*

Note that $T(n)$ is the sum of the nth powers of the eigenvalues of the matrix M (with the exception of $n = 0$, which is a special case because eigenvalues that equal 0 can mess things up via the problematical quantity 0^0). Thus the reciprocity formula $T(-n) = T(n)$ is equivalent to the assertion that the non-zero eigenvalues of the matrix, counted with multiplicity, come in reciprocal pairs (where 1 and -1, being their own reciprocals, are exempted from needing to be paired). And this is in fact true: the non-zero eigenvalues of the matrix are 1 (counted twice) and the two reciprocal roots of $t^2 - 4t + 1$.