Generating Functions **Math 421 - Workshop**

Definintion. A **probability generating function** (PGF) for a discrete random variable X is

$$E(t^X) = \sum_{k \in \text{support}(X)} t^k P(X = k)$$

where t is a symbolic variable (t doesn't represent a number).

Theorem. If X and Y are independent random variables with PGFs $g_X(t)$ and $g_Y(t)$ respectively, then the PGF for $X + Y$ is $g_X(t) \cdot g_Y(t)$.

1. Suppose you roll a six-sided die 6 times and add up the results. Find the probability that the total is 18.

2. I have a bag with a 4-sided, a 6-sided, an 8-sided, a 12-sided, and a 20-sided die. If I roll all five dice and add the results, what is the probability that the total is 12?

You can also use generating functions for counting instead of probability. For example, if you have 5 nickles, 2 dimes, and 1 quarter, how many different ways can you make 25 cents? You can multiply generating functions to find out:

$$\underbrace{(1 + x^5 + x^{10} + x^{15} + x^{20} + x^{25})}_{\text{nickles}} \underbrace{(1 + x^{10} + x^{20})}_{\text{dimes}} \underbrace{(1 + x^{25})}_{\text{quarters}}$$

3. Expand the expression above. How many ways are there to make 25 cents?

4. How many ways can \$500 be changed into \$1, \$5, \$10, and \$20 bills? That is, how many nonnegative integer solutions of $n_1 + 5n_5 + 10n_{10} + 20n_{20} = 500$ are there?

Using SymPy

To work with large polynomials, it helps to use a computer algebra system. I recommend the SymPy library for Python.

```python
from sympy import *
x, t = symbols("x, t")
```

Here are some things to know when using SymPy.

- You have to define which letters are symbolic variables using the `symbols()` function.
- Don't forget to use `*` for multiplication and `**` for powers.
- Use Python generator expressions to help construct really long polynomials:

  ```python
  # Do this:
  sum(x**k for k in range(11))

  # instead of this:
  1+x+x**2+x**3+x**4+x**5+x**6+x**7+x**8+x**9+x**10
  ```

- SymPy has built in functions `expand()` and `factor()` to expand and factor algebraic expressions. There are also functions to differentiate, integrate, and more. Here is sample code to expand the expression in problem 3:

  ```python
  nickles = sum(x**(5*k) for k in range(6))
  dimes = sum(x**(10*k) for k in range(3))
  quarters = sum(x**(25*k) for k in range(2))
  expand(nickles*dimes*quarters)
  ```

- Algebraic expressions in SymPy have a `coeff()` method, which can be useful when you have large polynomials. Here is an example of how to find the coefficient of x^{10} in a large polynomial. Notice that you have to expand the polynomial before using the `coeff()` method.

  ```python
  p = (1+x+x**2)**10
  print(expand(p).coeff(x**10)) # 8953
  ```