Algorithmen und Datenstrukturen im WS 2005/06

Polynom-Berechnung

Polynom-Berechnung

- Wie berechnet man Polynom
  - bei gegebenen Faktoren?

$$P_i(x) = P_{i-1}(x) \cdot f_{i-1}(x),$$

  - hier

$$P_0(x) = 1,\ f_{i-1}(x) = (c_i - x).$$

  - Allgemein gilt

$$P_0(x) = 1,$$
$$P_1(x) = P_0(x) \cdot (c_1 - x) = (c_1 - x),$$
$$P_2(x) = P_1(x) \cdot (c_2 - x) = (c_1 - x) \cdot (c_2 - x) = c_1 \cdot c_2 - (c_1 + c_2) \cdot x + x^2,$$
$$P_3(x) = P_2(x) \cdot (c_3 - x) = c_1 \cdot c_2 \cdot c_3 - (c_1 + c_2) \cdot c_3 \cdot x + c_3 \cdot x^2 - c_1 \cdot c_2 \cdot x + (c_1 + c_2) \cdot x^2 - x^3 =$$
$$= c_1 \cdot c_2 \cdot c_3 - (c_1 \cdot c_2 + c_1 \cdot c_3 + c_2 \cdot c_3) \cdot x + (c_1 + c_2 + c_3) \cdot x^2 - x^3,$$

Polynom-Berechnung

- Wie berechnet man Polynom
  - Multiplikation von Polynomen (U. G.)

```
public Polynom multiply (Polynom p) {
  Polynom ret = new Polynom ();
  for (int i = 0; i <= this.degree; i++) {
    for (int j = 0; j <= p.degree; j++) {
      ret = ret.add (i + j,
                     this.getCoeff (i) * p.getCoeff (j));
  } }
  ret.calcDegree ();
  return ret;
}
```

  - Adäquate Lösung

Polynom-Berechnung

- Wie berechnet man Polynom
  - Koeffizientenberechnung (S. G.)

```
double [] intcoeff;
double integralCoefficient(double[] zeroPoints,
                           int numberOfFactors, int from) {
  if(numberOfFactors==0) return 1;
  double sum = 0;
  for(int i=from;i<=zeroPoints.length-numberOfFactors;
          i++)
   sum += zeroPoints[i]*integralCoefficient(zeroPoints,
                                  numberOfFactors-1, i+1);
  return sum;
}
```

Polynom-Berechnung

- Wie berechnet man Polynom
  - Koeffizientenberechnung (S. G.)

```
// n=from
// numberOfFactors=1: c[n]*1 + c[n+1]*1 + ...
// numberOfFactors=2: c[n]*  ( c[n+1]+c[n+2] ..) +
                      c[n+1]*( c[n+2]+c[n+3] ..) +
                      c[n+2]*( c[n+3]+c[n+4] ..)
// numberOfFactors=3: c[n]*  ( c[n+1]*( c[n+2]+c[n+3] ..) +
                      c[n+2]*( c[n+3]+c[n+4] ..) +
                      c[n+3]*( c[n+4]+c[n+5] ..)
// numberOfFactors=4: c[n]*( c[n+1]*( c[n+3]*( c[n+4]+c[n+5]
                                                  ..) ..)
```

Polynom-Berechnung

- Wie berechnet man Polynom
  - Ableitung bei gegebenen Koeffizienten

```
double derivation(double coefficients[], int n, double x) {
  if(n>=coefficients.length-1) return -1;
  return -product(coefficients,n+1,x) +
         (c[n]-x)*derivation(coefficients,n+1,x);
}

double product(double coefficients[], int from, double x){
  // (c[n]-x)*(c[n+1]-x)* ...
  double sum = 1;
  for(int i=from;i<coefficients.length;i++)
    sum*=(coefficients[i]-x);
  return sum;
}
```

Polynom-Berechnung

- Wie berechnet man Polynom
  - Ableitung bei gegebenen Koeffizienten

```
double derivation(double coefficients[], int n, double x) {
  if(n>=coefficients.length-1) return -1;
  return -product(coefficients,n+1,x) +
         (c[n]-x)*derivation(coefficients,n+1,x);
}

// Produkt-Regel:
          (u*v)' = u'*v + u*v',
          u = (c-x), v = (d-x)*(e-x)*..
          u'= -1,    v'= (d-x)*(e-x)*..
```