Computerpraktikum Polyvalenz Ulrich Thiel
SS 2014

Nullstellen rationaler Polynome

Anforderungen:

- **(a) Bearbeiten Sie <u>alle</u> in dem Text gestellten Aufgaben.**
- **(b) Reichen Sie die implementierten Intrinsics ausführlich dokumentiert in <u>einer</u> Datei ein.**
- **(c) Schreiben Sie einen Text, in dem Sie die Aufgabenstellung kurz in eigenen Worten zusammenfassen, und Ihre Ideen, Ansätze und Ergebnisse erläutern.**
- **(d) Bereiten Sie einen Vortrag (15-20 Minuten) darüber vor.**
- **(e) Sind Ergebnisse zu den Aufgaben bereits in der Literatur diskutiert (und das sind sie meistens), möchte ich keine Referenz auf diese Literatur als Lösung bekommen, sondern funktionsfähige Algorithmen (bzw. Beweise), die diese Lösungen ergeben.**
- **(f) Sie dürfen alle in MAGMA bereits implementierten Intrinsics benutzen, es sei denn, es ist ausdrücklich untersagt.**

Angenommen, wir haben ein rationales Polynom wie

$$x^{18} - x^{16} + 2x^{15} - x^{14} - x^5 + x^4 + x^3 - 3x^2 + 3x - 1 \in \mathbb{Q}[x] \tag{1}$$

und möchten wissen, wie viele positive/negative reelle Nullstellen bzw. wie viele Paare komplex-konjugierter echt komplexer Nullstellen es besitzt. Wir können wir dieses Problem lösen?

1. Aufgabe. Es gibt Algorithmen, die die komplexen Nullstellen eines rationalen Polynoms *approximieren* können, wie z.B. den Schönhage-Algorithmus. Dieser ist auch in MAGMA implementiert (Intrinsic Roots für reelle bzw. komplexe Polynome). Verwenden Sie diesen Ansatz, um eine Intrinsic

ApproximateRootClassification(f::RngUPolElt) -> Tup

zu implementieren, die ein Tupel $((p, n, z), c)$ zurückgibt, wobei

- $p = (p_1, \ldots, p_m)$ eine Liste ist, die die verschiedenen positiven Nullstellen mit Multiplizität kodiert, d.h. es gibt m positive Nullstellen mit Multiplizitäten p_i.
- n eine analoge Liste für die negativen Nullstellen ist.
- z die Multiplizität der 0 als Nullstelle ist.
- $c = (c_1, \ldots, c_r)$ eine Liste ist, die die komplex-konjugierten Paare echt komplexer Nullstellen kodiert, d.h. es gibt r solche Paare mit Multiplizitäten c_i.

Zum Beispiel:

```
> P<x> := PolynomialRing(Rationals());
> f:=x^18 - x^16 +2*x^15 - x^14 - x^5 +x^4 +x^3 -3*x^2 +3*x - 1;
> ApproximateRootClassification(f);
<<[ 1 ], [ 1 ], []>, [ 2, 1, 1, 1, 1, 1, 1 ]>
>
```

Es gibt also nur eine positive Nullstelle (mit Multiplizität 1), eine negative Nullstelle (mit Multiplizität 1), die 0 ist keine Nullstelle und es gibt 7 komplex-konjugierte Paare echt-komplexer Nullstellen, eine mit Multiplizität 2, alle andere mit Multiplizität 1.

2. Aufgabe. Im obigen Ansatz werden die Nullstellen approximiert – standardmäßig auf 30 Nachkommastellen. Das ist natürlich ein Problem, denn verschiedene sehr nah beieinander liegende Nullstellen könnten dadurch unter Umständen als gleich angesehen werden, sodass das Ergebnis von `ApproximateRootClassification` schlicht *falsch* ist. Demonstrieren Sie dies an einem Beispiel.

3. Aufgabe. Überraschenderweise[1] gibt es für das Problem eine rein algebraische und damit korrekte Methode, die in [1] (Abschnitt "An algorithm for root classification") abstrakt dargestellt wird. Implementieren Sie eine Intrinsic `RootClassification` analog zu `ApproximateRootClassification`, die diese Methode verwendet um die Nullstellen eines rationalen Polynoms zu klassifizieren. Zeigen Sie, dass Ihr Gegenbeispiel von oben damit korrekt behandelt wird.

Literatur

[1] Yang. *Recent advances on determining the number of real roots of parametric polynomials*. In: *J. Symbolic Computation* 28 (1999), S. 225–242.

[1] Für mich zumindest