# Kleine Einfache Gruppen

Patrick Dabbert, Stephan Hilb und Martin Rösner

15. Februar 2013

# Die Aufgabenstellung

**Aufgabe 1:**
*Fertigen Sie eine Liste der endlichen einfachen nicht-abelschen Gruppen bis Ordnung 10.000 an. Sie dürfen hierbei verwenden, dass diese in folgender Liste enthalten sind:* $A_n, PSL(2,q), PSL(3,3), PSU(3,3)$ *und* $M_{11}$. *Verwenden sie, dass die Ordnung eines endlichen Körpers eine Primzahlpotenz ist. Welche der von ihnen gefundenen Gruppen sind minimal einfach?*

# Definition: Einfache Gruppen

*Eine Gruppe heißt einfach wenn sie nur 1 und sich selbst als Normalteiler besitzt.*

# Definition Minimal Einfache Gruppen

*Eine einfache Gruppe G heißt minimal einfach, wenn alle echten Untergruppen von G auflösbar sind.*

# Lösungsschritte

1. *gegebene Gruppen in Liste eintragen*
2. *Überprüfen ob Gruppen Einfach, Abelsch und Ordnung kleiner 10000.*
3. *Gruppen auf Isomorphie prüfen*
4. *Untergruppen auf Auflösbarkeit prüfen (hierbei reicht es die die Konjugationsklassen der Untergruppen zu betrachten)* $\Rightarrow$ *Minimal einfach*

# Implementierung in GAP (Liste01.g)

```
#liste01.g
Liste01 := function ()
local Liste, Liste01, Liste02;
Liste := [];
#Schleife ueber die Ordnung
for n in [1..10] do
    #Alternierende Gruppe erstellen
    Add(Liste, AlternatingGroup(n));
od;
for n in [2..30] do
    # ist q primzahlpotenz? (Koerpergrade sind Primzahlpotenzen)
    if IsPrimePowerInt(n) then
        Add(Liste, PSL(2,n));
    fi;
od;
Add(Liste, PSL(3,3));
Add(Liste, PSU(3,3));
Add(Liste, MathieuGroup(11));

#Eigenschaften fuer einfache Gruppen ueberpruefen.
Liste := Filtered(Liste, g -> (IsSimpleGroup(g) and (not IsAbelian(g)) and Order(g)<= 10000))
```

# Implementierung in GAP (Liste01.g)

```
#liste01.g
#Isomorphe Gruppen ausschliessen
k := Size(Liste);
u := IsomorphismGroups(AlternatingGroup(3),AlternatingGroup(4));
Liste01 := [];
for i in [1..k] do
    p := 0;
    for j in [i+1..k] do
        if not u = IsomorphismGroups(Liste[i], Liste[j]) then
            p := 1;
        fi;
    od;
    if p = 0 then
        Add(Liste01, Liste[i]);
    fi;
od;
Liste02 := List(Liste01, l -> StructureDescription(l));

return Liste01;

end;
```

# Implementierung in GAP (gapfile01.g)

```
Read("liste01.g");
Liste01 := Liste01();
Liste02 := List(Liste01, l -> StructureDescription(l));
Print("Liste aller endlichen, einfachen, nicht-abelschen Gruppen bis Ordnung 10000: \n");
Print(Liste02, "\n");


Liste03:=[];
# minimal einfach? (alle untergruppen aufloesbar):
for G in Liste01 do
    minimal := 1;
    for U in List(ConjugacyClassesSubgroups(G), Representative) do
        if not IsSolvableGroup(U) and not U = G then
            minimal := 0;
            break;
        fi;
    od;
    if minimal = 1 then
        Add(Liste03, G);
    fi;
od;
Print("\nMinimale einfache Gruppen \n");
Liste04 := List(Liste03, l -> StructureDescription(l));
Print(Liste04, "\n");
```

# Ergebnis

| | |
|---|---|
| einfach | [ "A7", "A5", "PSL(3,2)", "PSL(2,8)", "A6", "PSL(2,11)", "PSL(2,13)", "PSL(2,16)","PSL(2,17)", "PSL(2,19)", "PSL(2,23)","PSL(2,25)", "PSL(2,27)", "PSL(3,3)", "PSU(3,3)","M11" ] |
| minimal einfach | [ "A5", "PSL(3,2)", "PSL(2,8)", "PSL(2,13)", "PSL(2,17)","PSL(2,23)", "PSL(2,27)", "PSL(3,3)" ] |

**Anmerkung**: PSL(3,2)$\cong$ PSL(2,7)

# Die Aufgabenstellung

**Aufgabe 2:**
*Zeigen Sie, dass $A_8$ und $PSL(3,4)$ nicht isomorph sind.*

# Lösungsschritte

*Überprüfe für $A_8$ und $PSL(3,4)$ für welche Ordnungen Elemente vorkommen.*

# Implementierung in GAP (gapfile02.g)

```
G := AlternatingGroup(8);
H := PSL(3,4);
Print("G = ", G, " ", Order(G), "\nH = ", H, " ", Order(G), "\n");
Print("Sind G und H isomorph? \n", IsomorphismGroups(G, H), "\n");
Liste01 := []; Liste02 := [];
for x in G do
    if not Order(x) in Liste01 then
        Add(Liste01, Order(x));
    fi;
od;
for x in H do
    if not Order(x) in Liste02 then
        Add(Liste02, Order(x));
    fi;
od;
Sort(Liste01); Sort(Liste02);
Print("In G gibt es Elemente der Ordnung: ", Liste01, "\n","In H gibt es Elemente der Ordnung
: ", Liste02,  "\n");
```

# Ergebnis

**Ausgabe***:*

```
G = AlternatingGroup( [ 1 .. 8 ] ) 20160
H = Group( [ ( 3, 4, 5)( 7, 9, 8)(10,14,18)(11,17,20)(12,15,21)(13,16,19),
 ( 1, 2, 6, 7,11, 3,10)( 4,14, 8,15,16,20,13)( 5,18, 9,19,21,17,12) ] ) 20160
Sind G und H isomorph?
fail
In G gibt es Elemente der Ordnung: [ 1, 2, 3, 4, 5, 6, 7, 15 ]
In H gibt es Elemente der Ordnung: [ 1, 2, 3, 4, 5, 7 ]
```

*Es gibt also in $A_8$ Elemente von Ordnung 6 und 15, aber nicht in PSL(3,4).* $\Rightarrow A_8 \ncong PSL(3,4)$

# Die Aufgabenstellung

**Aufgabe 3**:
*(In Zusammenarbeit mit Partitionen-Gruppe)*
*Bestimmen Sie, welche der von ihnen bestimmten Gruppen eine nicht-triviale Partition von Untergruppen besitzt.*

# Lösungsschritte

1. *Benutze erste Liste aus Aufgabe 1*
2. *Überprüfe für jede Gruppe ob nicht Triviale Partitionen vorliegen*
3. *Wenn ja, speichere die Gruppe in eine Liste*

# Implementierung in GAP (gapfile03.g)

```
Read("liste01.g");
Read("HasNonTrivialPartition.txt");
Read("FindAllPartitions.txt");
Liste01 := Liste01();

Liste03 := [];
Print("Liste aller endlichen, einfachen, nicht-abelschen Gruppen bis
 Ordnung 10000 mit nicht trivialen Partitionen");
for i in [1..Length(Liste01)] do
    G:=Liste01[i];
    if  HasNonTrivialPartition(G) then
        Add(Liste03, G);
        Print(StructureDescription(G), " hat nicht triviale Partitionen \n");
    fi;
od;
```

# Ergebnis

*Programm rechnet noch, Zwischenergebnisse bis PSL(3,3):*
*Mit nicht trivialen Partitionen: A5, PSL(3,2), PSL(2,8), A6 PSL(2,11), PSL(2,13), PSL(2,16), PSL(2,17), PSL(2,19), PSL(2,23), PSL(2,25), PSL(2,27)*
*ohne: A7*
*noch nicht getestet: PSL(3,3), PSU(3,3), M11*

# Die Aufgabenstellung

**Aufgabe 4:**

*(In Zusammenarbeit mit Sylowgruppen-Gruppe) Bestimmen Sie, welche der von Ihnen gefundenen Gruppen N-Gruppen sind. Geben Sie außerdem für* $P \in Syl_2(G)$ *und* $P \in Syl_3(G)$ *die Isomorphietypen von* $P$*,* $Z(P)$*,* $C_G(P)$ *und* $N_G(P)$ *an, wobei* $G$ *eine Gruppe aus ihrer Liste ist.*

# Lösungsschritte

1. Benutze erste Liste aus Teil eins.
2. Überprüfe die Gruppen darauf ob sie N Gruppen sind
3. wenn ja: Gib Isomorphietyp , Z(P), $C_G(P)$ und $N_G(P)$ an.

# Implementierung in GAP (gapfile04.g)

```
Read("liste01.g");
Read("IsNGroup.g");
Read("StructureCentralizerSylowSubgroup.g");
Read("StructureCentreSylowSubgroup.g");
Read("StructureNormalizerSylowSubgroup.g");
Read("StructureSylowSubgroup.g");
Liste01 := Liste01();
Liste03 := [];
Print("Liste aller N-Gruppen davon: \n");
for G in Liste01 do
    if IsNGroup(G) then
        Print("==== ", StructureDescription(G), " ====\n");
        Print("Syl2G-isomorphietyp: ", StructureSylowSubgroup(2, G), "\n");
        Print("Syl3G-isomorphietyp: ", StructureSylowSubgroup(3, G), "\n");
        Print("Zentrum: ", StructureCentreSylowSubgroup(2, G), "\n");
        Print("Zentralisator ", StructureCentralizerSylowSubgroup(2, G), "\n");
        Print("Normalisator ", StructureNormalizerSylowSubgroup(2, G), "\n\n");
        Add(Liste03, G);
    fi;
od;
Liste04 := List(Liste03, l -> StructureDescription(l));
Print(Liste04, "\n");
```

# Ergebnis

*Alle getesteten Gruppen sind N-Gruppen. (Genaueres siehe Tabelle)*

| | |
|---|---|
| einfach | [ "A7", "A5", "PSL(3,2)", "PSL(2,8)", "A6", "PSL(2,11)", "PSL(2,13)", "PSL(2,16)","PSL(2,17)", "PSL(2,19)", "PSL(2,23)","PSL(2,25)", "PSL(2,27)", "PSL(3,3)", "PSU(3,3)","M11" ] |
| minimal einfach | [ "A5", "PSL(3,2)", "PSL(2,8)", "PSL(2,13)", "PSL(2,17)","PSL(2,23)", "PSL(2,27)", "PSL(3,3)" ] |
| N-Gruppen | [ "A7", "A5", "PSL(3,2)", "PSL(2,8)", "A6", "PSL(2,11)", "PSL(2,13)", "PSL(2,16)","PSL(2,17)", "PSL(2,19)", "PSL(2,23)","PSL(2,25)", "PSL(2,27)", "PSL(3,3)", "PSU(3,3)","M11" ] |