Verteilte Systeme 2012: 7. Übungszettel

Max Michels
Schintke, Schütt Philipp Borgers
21.06.2012 Sascha Schönfeld

1 Maekawa-Algorithmus in Erlang

1.a Code

```
-module(maekawa).
-export([init/0, initProc/0]).

%% feel free to adjust the init function

init() ->
    %%6 processes and 3 groups
    P1 = spawn(maekawa, initProc, []),
    P2 = spawn(maekawa, initProc, []),
    P3 = spawn(maekawa, initProc, []),
    P4 = spawn(maekawa, initProc, []),
    P5 = spawn(maekawa, initProc, []),
    P6 = spawn(maekawa, initProc, []),
    G1 = [P1,P2,P3],
    G2 = [P2,P4,P5],
    G3 = [P5,P6,P1],
    P1 ! {setGroup, G1},
    P2 ! {setGroup, G2},
    P3 ! {setGroup, G1},
    P4 ! {setGroup, G2},
    P5 ! {setGroup, G3},
    P6 ! {setGroup, G3},
    P1 ! enterCriticalSection,
    P3 ! enterCriticalSection,
    P6 ! enterCriticalSection,
    'initialization done'.

%%tell a process which group it belongs to
initProc() ->
    InitState = {_State = released, _Voted = false, _Requests = [], _Replies = 0},
    receive
        {setGroup, V} ->
            io:format("initializing ~p with ~p~n",[self(), V]),
            loop(InitState, V);
        _ ->
            io:format("init for ~p failed~n",[self()])
    end.

%%main loop
loop(State, V) ->
    {ProcState, Voted, Requests, Replies} = State,
    NewState =
        case ProcState of
            held ->
                %%we can now safely enter the critical sections
                criticalSection(),
                [GroupMember ! release || GroupMember <- V],
                {_ProcState = released, Voted, Requests, Replies};
            wanted ->
                if
                    Replies =:= length(V) ->
                        {_ProcState = held, Voted, Requests, 0};
                    true ->
                        processMessages(State, V)
                end;
            released ->
                processMessages(State, V)
        end,
    %io:format("~p: state is ~p~n",[self(), NewState]),
    loop(NewState, V).

%%message processing
processMessages(State, V) ->
    {ProcState, Voted, Requests, Replies} = State,
        receive
            enterCriticalSection ->
                io:format("~p: trying to enter critical section~n",[self()]),
                [GroupMember ! {request, self()} || GroupMember <- V],
                io:format("~p: sent out requests to group ~n", [self()]),
                {_State = wanted, _Voted = false, Requests, Replies};
            {request, Pid} ->
                case State of
                    {_ProcState = held, _Voted, _Requests, _Replies} ->
                        io:format("~p: queuing request from ~p~n",[self(), Pid]),
                        {ProcState, Voted, Requests ++ [Pid], Replies};
                    {_ProcState, _Voted = true, _Requests, _Replies} ->
                        io:format("~p: queuing request from ~p~n",[self(), Pid]),
                        {ProcState, Voted, Requests ++ [Pid], Replies};
                    _ ->
                        io:format("~p: Sending reply to ~p~n", [self(), Pid]),
                        Pid ! reply,
                        {ProcState, _Voted = true, Requests, Replies}
                end;
            release ->
                case Requests of
                    [] ->
                        {ProcState, _Voted = false, [], Replies};
                    [H | T] ->
                        io:format("~p: Sending reply to ~p~n", [self(), H]),
                        H ! reply,
                        {ProcState, _Voted = true, T, Replies}
                end;
            reply ->
                io:format("~p: Received reply number ~p~n", [self(), Replies+1]),
                {ProcState, Voted, Requests, Replies + 1};
            _ ->
                io:format("~p: WARNING i don't understand this message: ~~n",[self()]),
                State
        end.

%%the critical section
criticalSection() ->
    io:format("~p: I'm in the critical section. Hope noone's around... ~n", [self()]),
    timer:sleep(1000),
    io:format("~p: I'm leaving the critical section~n", [self()]).
```

1.b Erläuterung

Wir haben den Maekawa-Algorithmus für eine flexible Anzahl von Prozessen implementiert. In der Funktion init() können beliebig viele Prozesse gespawnt werden. Damit der wechselseitige Ausschluss zwischen den Prozessen funktioniert, muss jedem Prozess eine Prozessgruppe zugeordnet werden. Dies kann nach der Initialisierung des Prozesses mit eine Nachricht setGroup, GROUP an den jeweiligen Prozess erfolgen. Anschließend können Prozesse mit einer Nachricht der Form enterCriticalSection in den kritischen Bereich geschickt werden. Der kritische Bereich wird von der Funktion criticalSection() simuliert. Der Maekawa-Algorithmus stellt in der Funktion loop() sicher, dass stets nur ein Prozess im kritischen Bereich landet. Das heißt, dass wenn ein Prozess den Zustand held hat, die anderen Prozesse warten müssen und entsprechende Anfragen gebuffert werden. Für einen Prozess gibt es drei Zustände, in dem er sein kann. Entweder ist ein prozess in der kritischen Sektion (held), oder er wartet auf eine Freigabe (wanted), oder aber er befindet sich im normalen Ablauf (released). Im Zustand held und wanted werden stets Nachrichten abgeareitet, wohingegen im held Zustand nicht. Der eigentliche Algorithmus sieht auch im Held Zustand eine Abarbeitung der Nachrichten vor, jedoch führt unsere Variante auch zum wechselsetitigen Ausschluss.

Verteilte Systeme 2012: 7. Übungszettel

Max Michels
Schintke, Schütt Philipp Borgers
21.06.2012 Sascha Schönfeld

1.c Testlauf

```
maekawa:init().
initializing <0.156.0> with [<0.156.0>,<0.157.0>,<0.158.0>]
initializing <0.157.0> with [<0.157.0>,<0.159.0>,<0.160.0>]
initializing <0.158.0> with [<0.156.0>,<0.157.0>,<0.158.0>]
initializing <0.159.0> with [<0.157.0>,<0.159.0>,<0.160.0>]
initializing <0.160.0> with [<0.160.0>,<0.161.0>,<0.156.0>]
initializing <0.161.0> with [<0.160.0>,<0.161.0>,<0.156.0>]
'initialization done'
<0.156.0>: trying to enter critical section
<0.158.0>: trying to enter critical section
<0.158.0>: sent out requests to group
<0.161.0>: trying to enter critical section
<0.156.0>: sent out requests to group
<0.157.0>: Sending reply to <0.158.0>
<0.158.0>: Sending reply to <0.158.0>
<0.158.0>: queuing request from <0.156.0>
<0.161.0>: sent out requests to group
<0.158.0>: Received reply number 1
<0.156.0>: Sending reply to <0.158.0>
<0.157.0>: queuing request from <0.156.0>
<0.158.0>: Received reply number 2
<0.160.0>: Sending reply to <0.161.0>
<0.161.0>: Sending reply to <0.161.0>
<0.156.0>: queuing request from <0.156.0>
<0.158.0>: Received reply number 3
<0.158.0>: I'm in the critical section. Hope noone's around...
<0.156.0>: queuing request from <0.161.0>
<0.161.0>: Received reply number 1
<0.161.0>: Received reply number 2
<0.158.0>: I'm leaving the critical section
<0.158.0>: Sending reply to <0.156.0>
<0.156.0>: Sending reply to <0.156.0>
<0.156.0>: Received reply number 1
<0.156.0>: Received reply number 2
<0.157.0>: Sending reply to <0.156.0>
<0.156.0>: Received reply number 3
<0.156.0>: I'm in the critical section. Hope noone's around...
<0.156.0>: I'm leaving the critical section
<0.156.0>: Sending reply to <0.161.0>
<0.161.0>: Received reply number 3
<0.161.0>: I'm in the critical section. Hope noone's around...
<0.161.0>: I'm leaving the critical section
```