Efficient Built-in Dynamic Optimization Features of OpenModelica

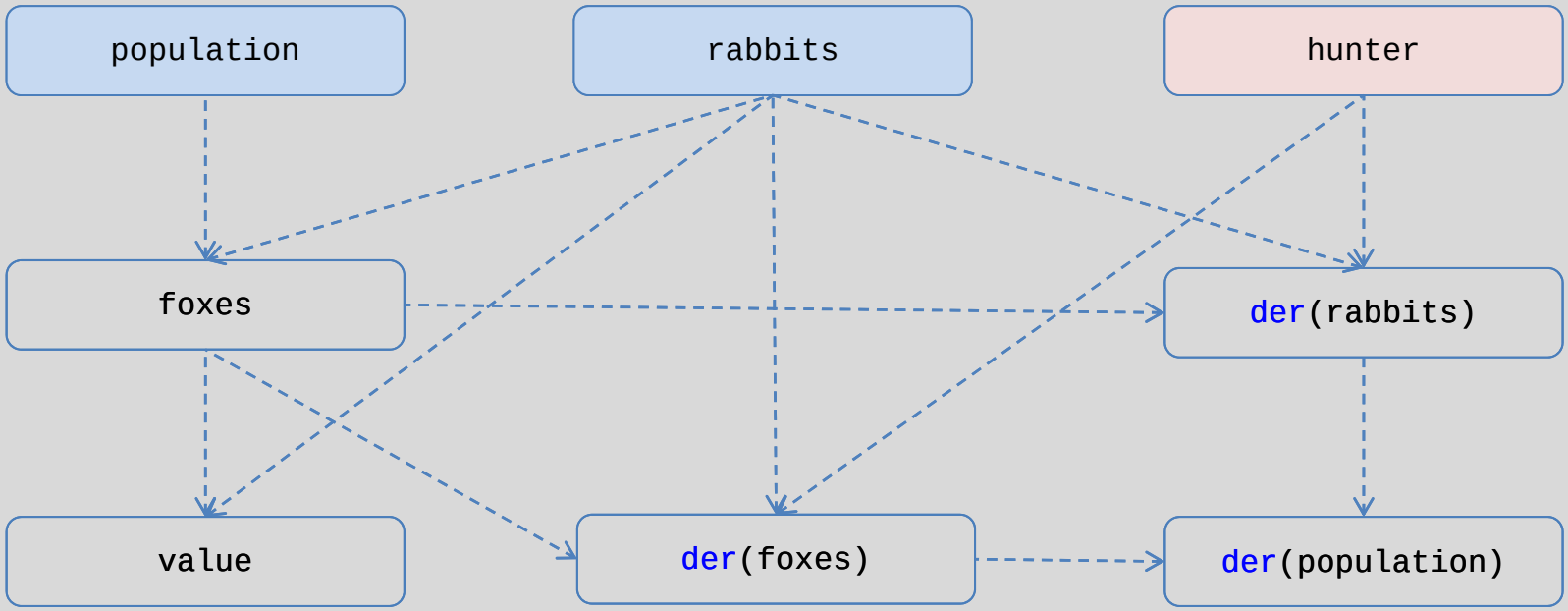
Vitalij Ruge

Bernhard Bachmann

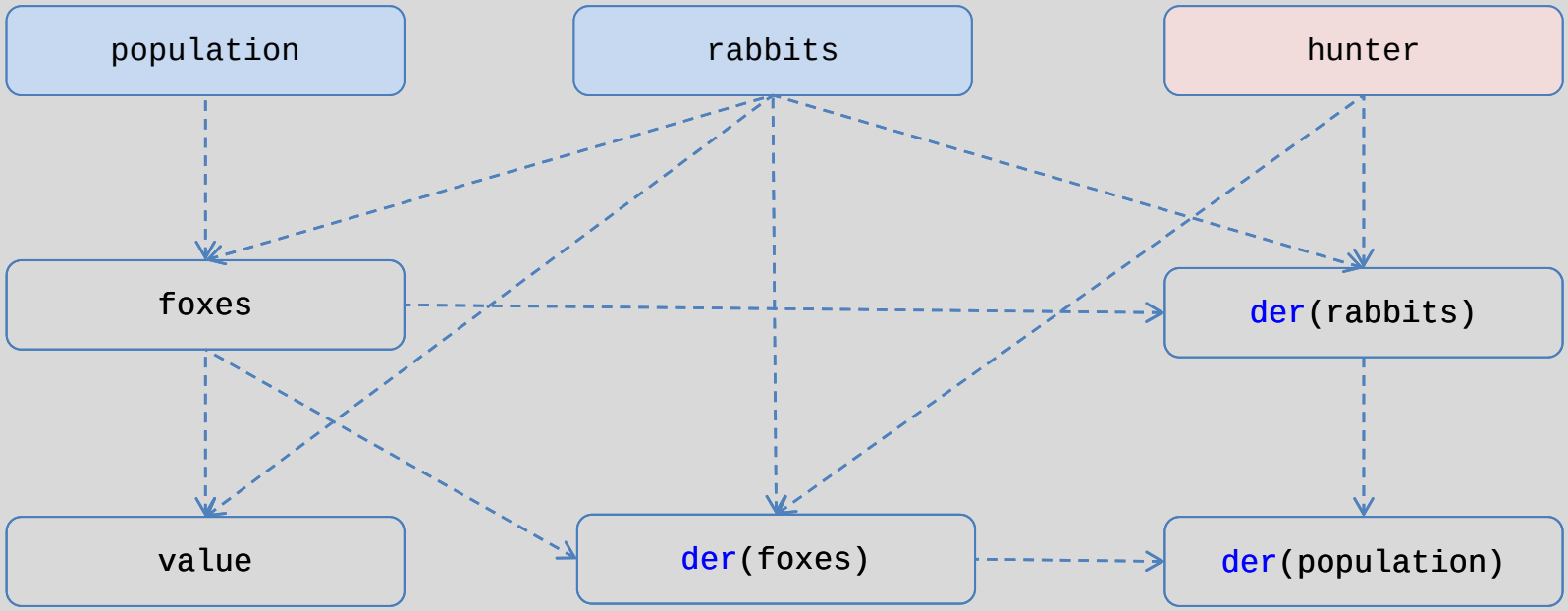
Example - Simulation

```
model forest
  Real foxes(min=0, max=100);
  Real rabbits(min=0, max=1000);
  Real population(start=850, min=550, max=1100, stateSelect=StateSelect.always);
  Real value;

  parameter Real g_r=0.4, d_rf=0.005;
  parameter Real d_f=0.9, g_fr=1.0;
  parameter Real priceFox=150, priceRabbit=12;
  input Real hunter(min=0, max=1, start=0.5);

initial equation
  der(foxes) = 20;
  value = 11000;

equation
  der(rabbits) =  rabbits*g_r - rabbits*foxes*d_rf - hunter*rabbits*0.01;
  der(foxes) = -foxes*d_f + rabbits*foxes*d_rf*g_fr - hunter*foxes*2;
  population = foxes+rabbits;
  value = priceFox*foxes + priceRabbit*rabbits;
end forest;

optimization forestOpt(objectiveIntegrand=-value)
  extends forest;
end forestOpt;
```

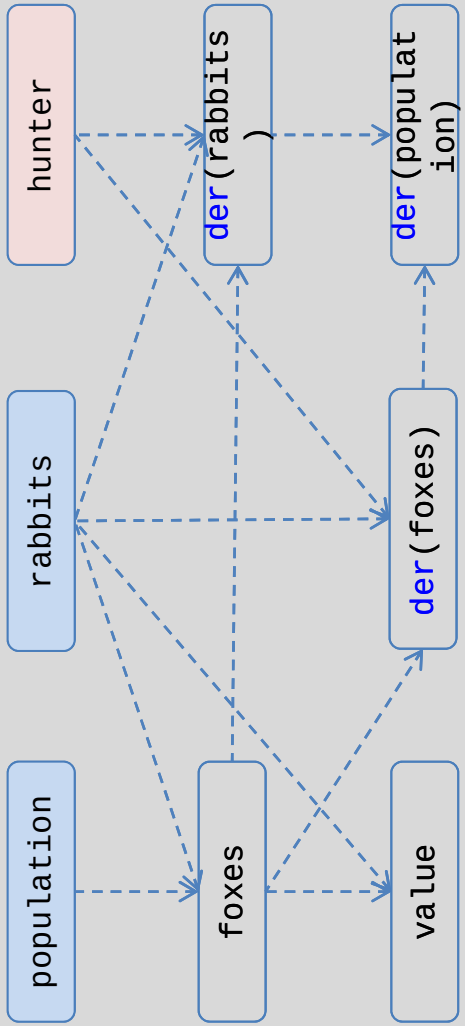
Example – Simulation Dependence Graph



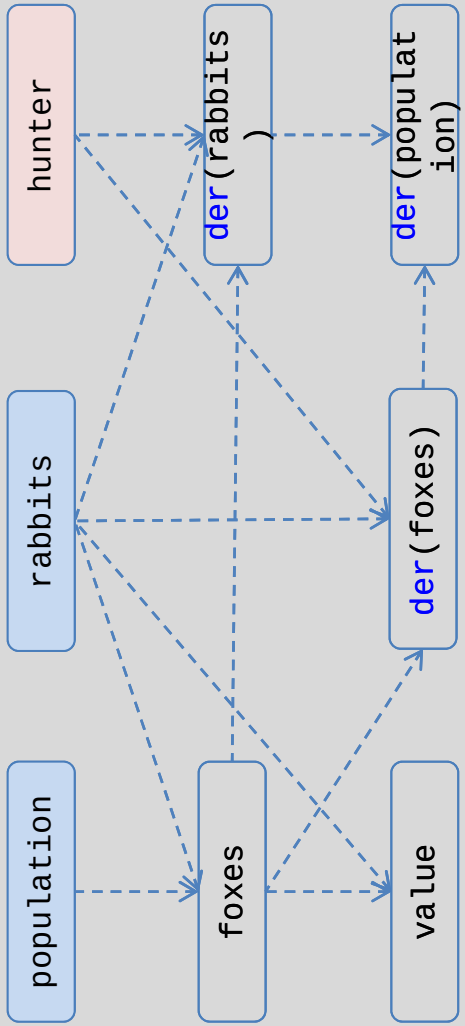
Example – integration step



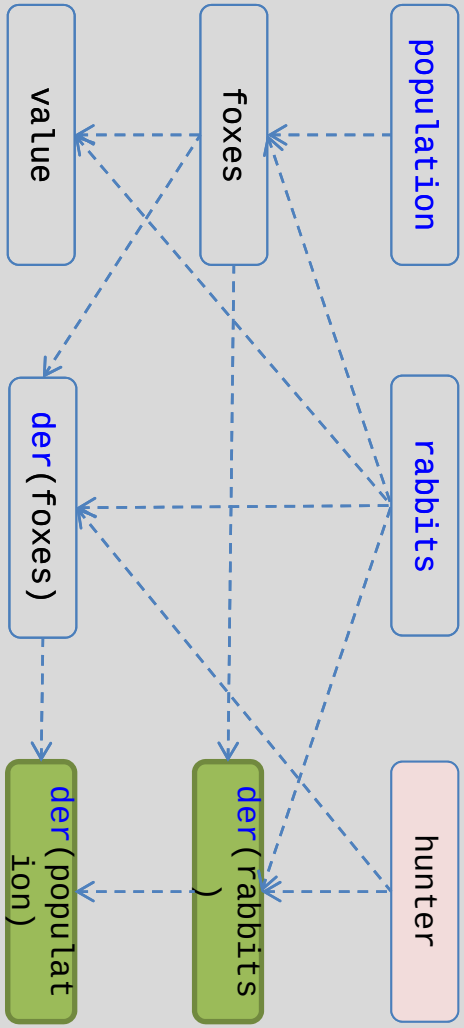
Example – integration step



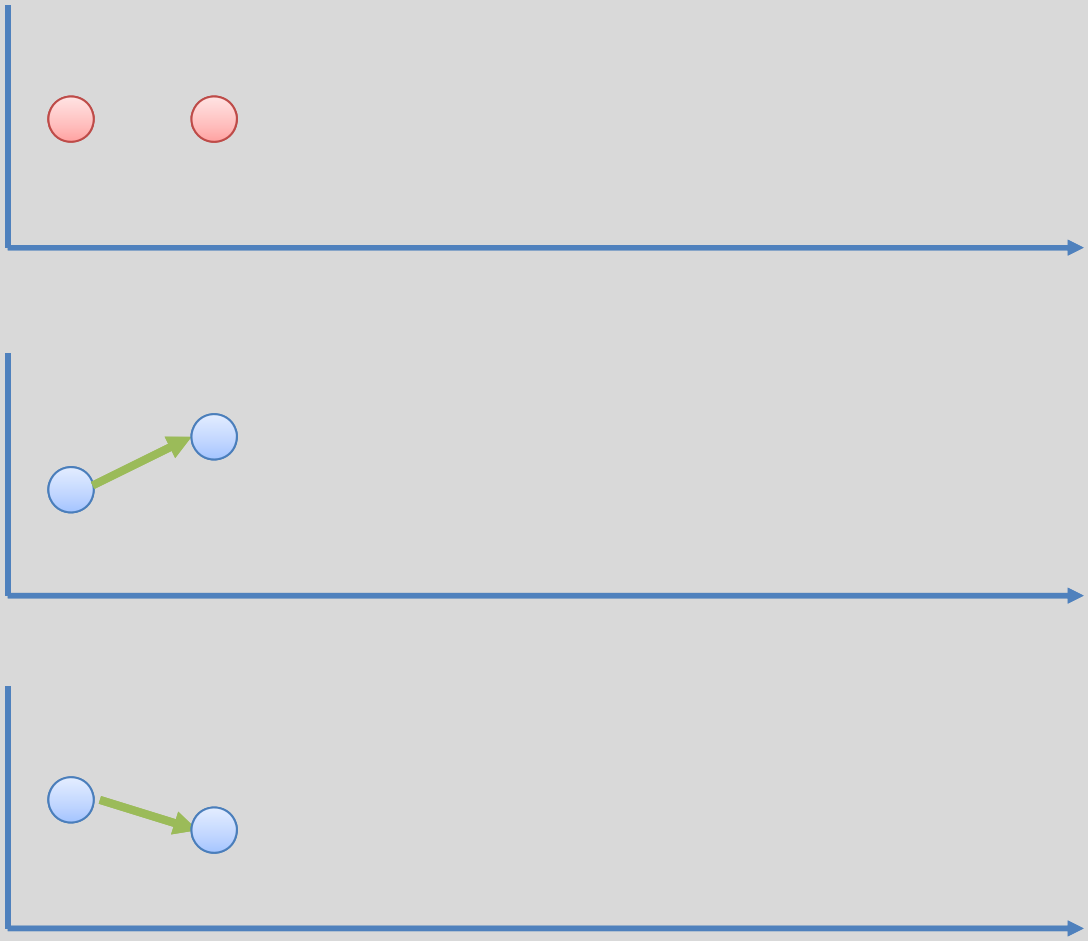
Example – integration step

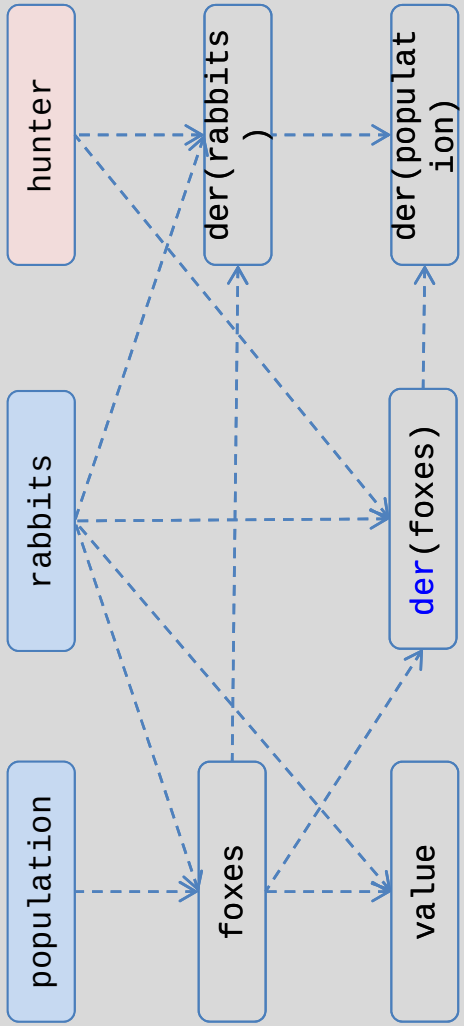


Example – integration step

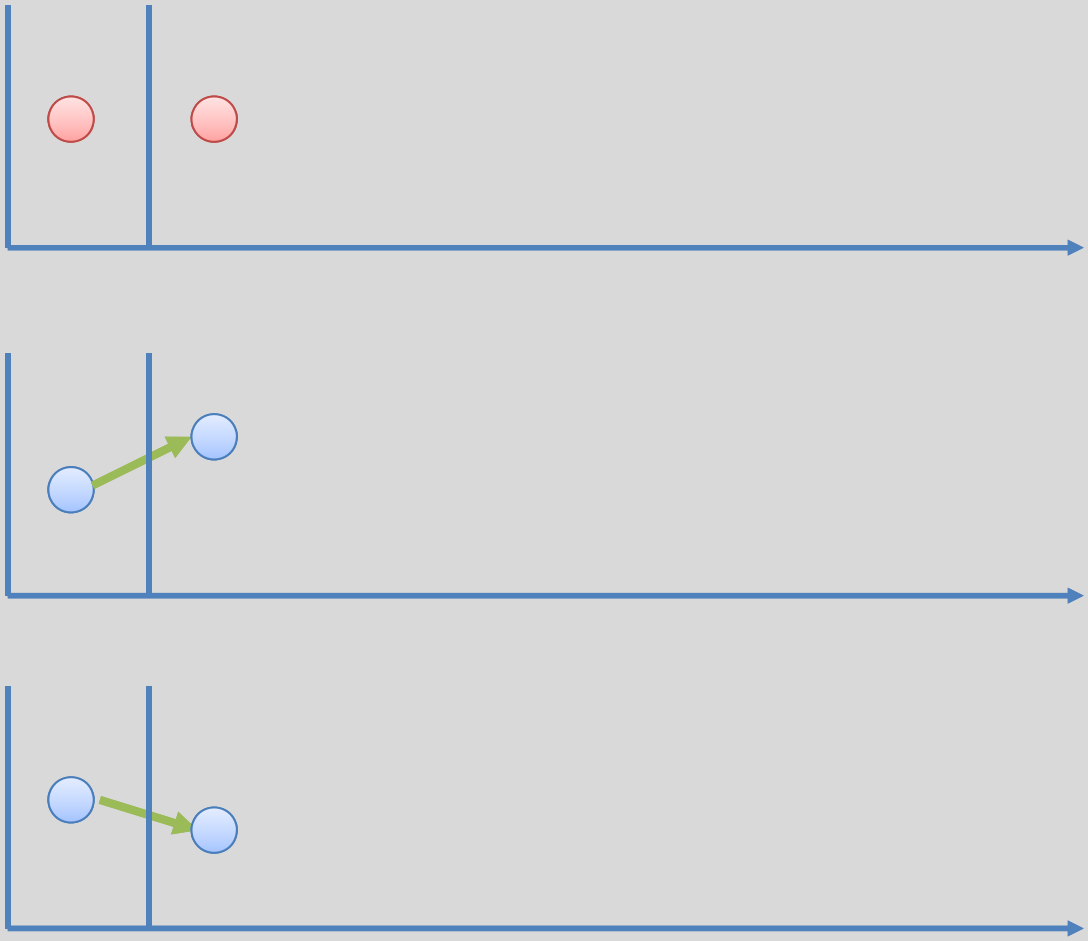


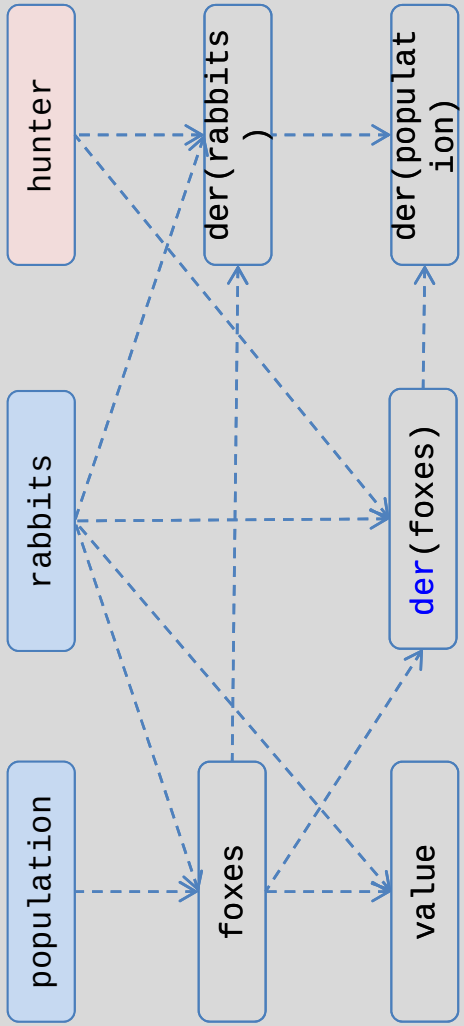
Example – integration step



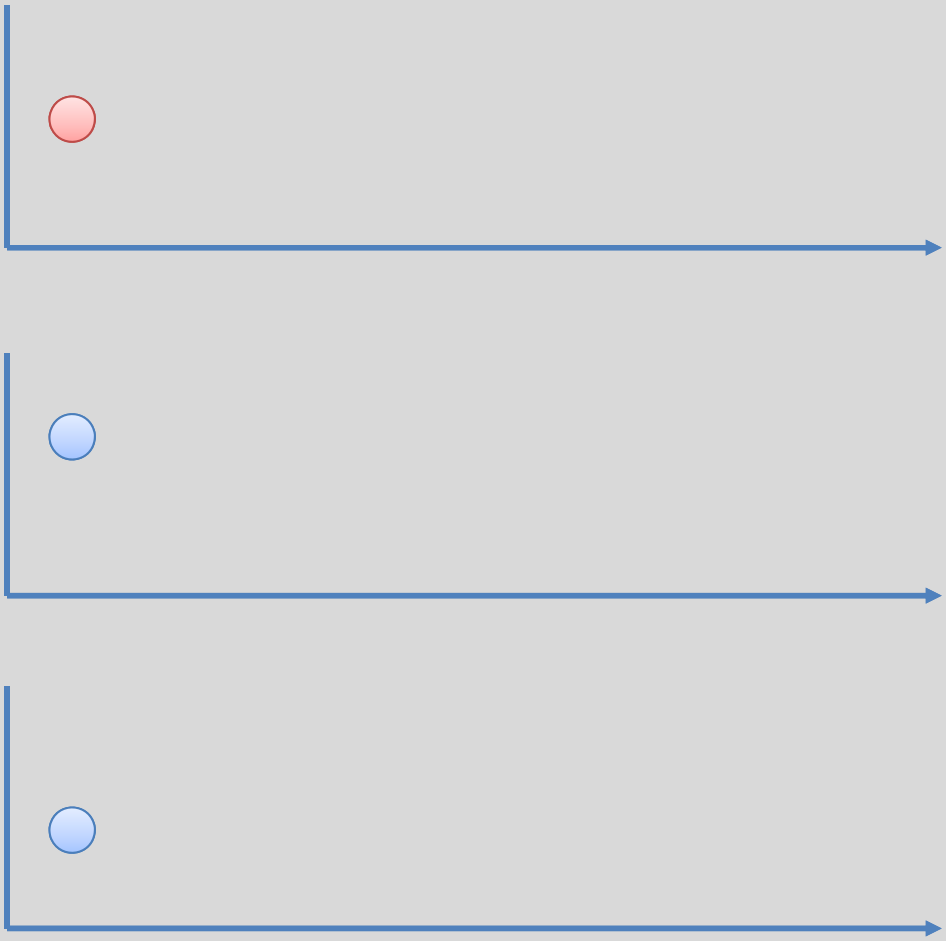


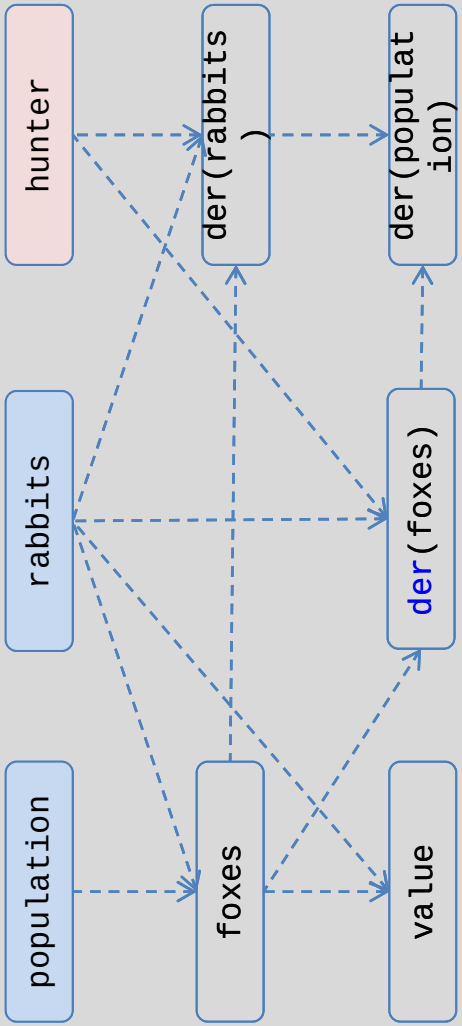
Example – integration step



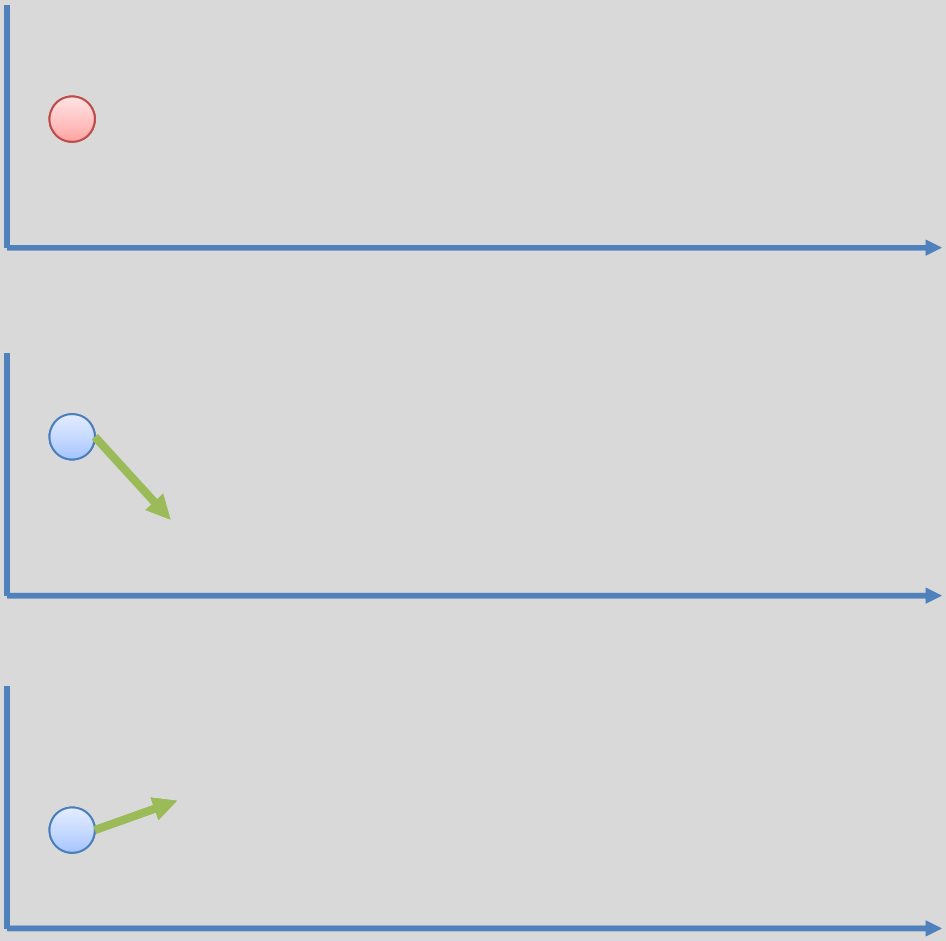


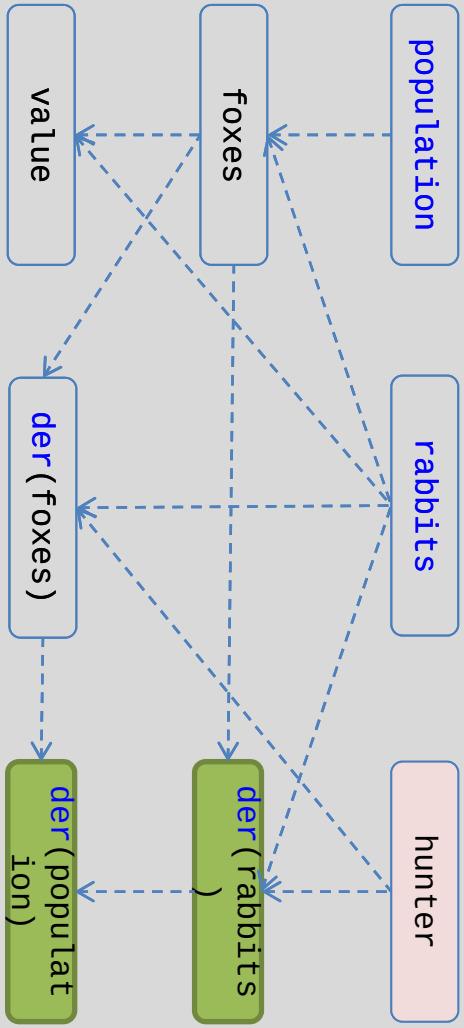
Example – integration step



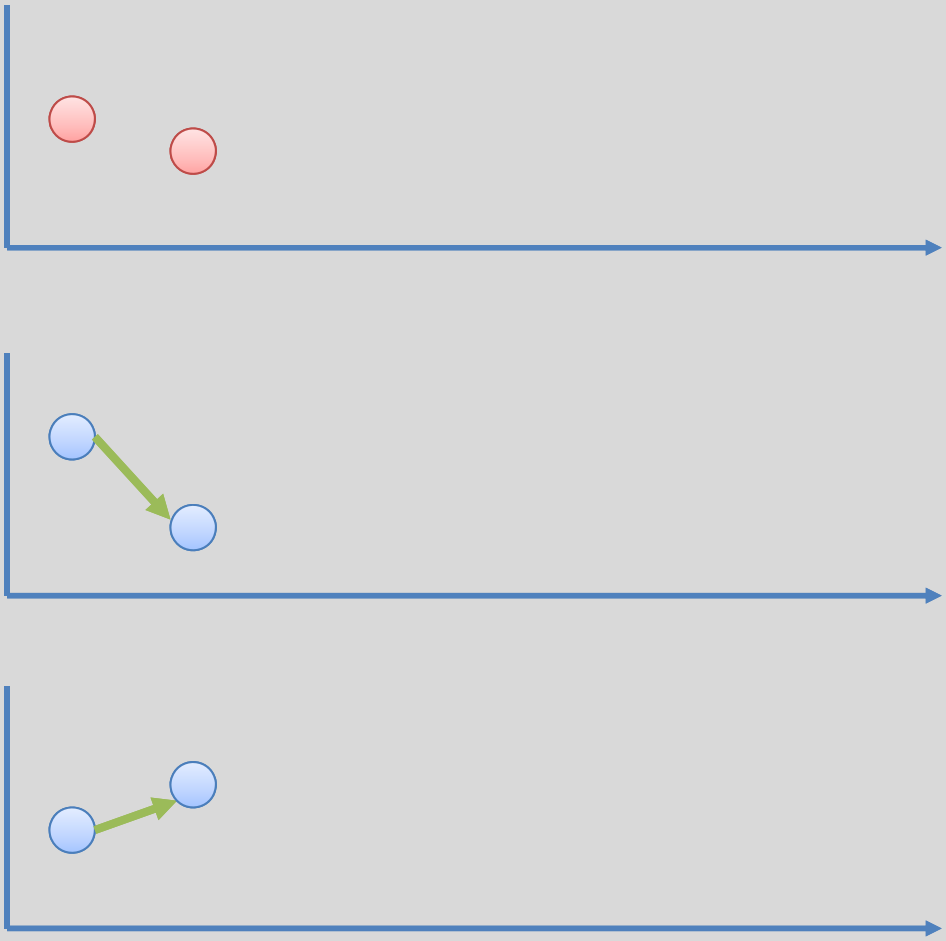


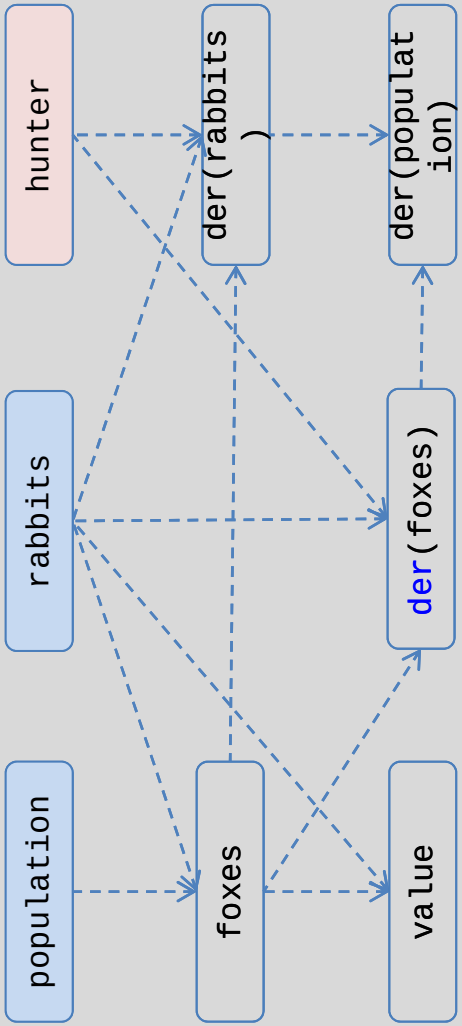
Example – integration step





Example – integration step

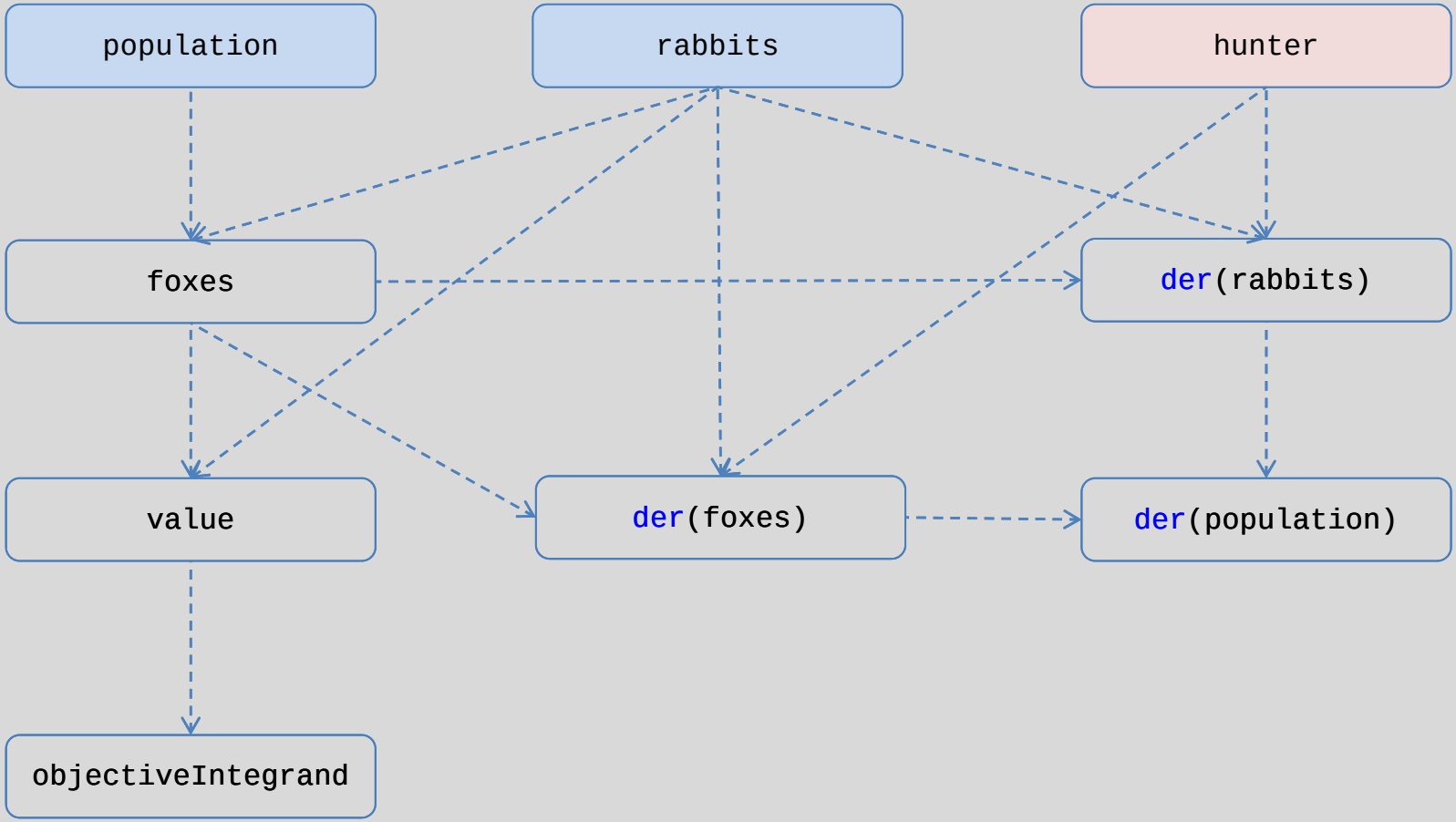




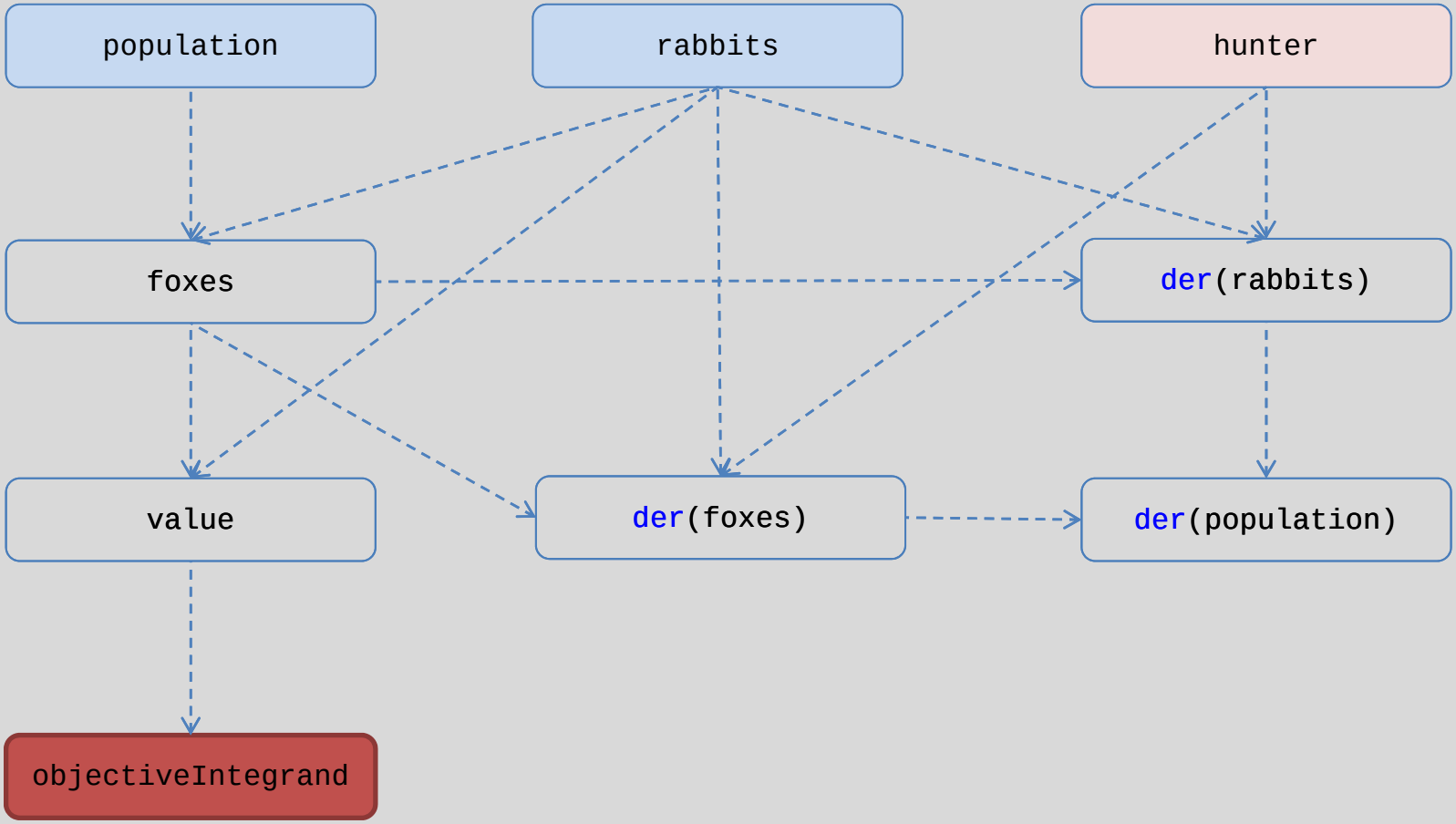
Example - Optimization

```
model forest
  Real foxes(min=0, max=100);
  Real rabbits(min=0, max=1000);
  Real population(start=850, min=550, max=1100, stateSelect=StateSelect.always);
  Real value;

  parameter Real g_r=0.4, d_rf=0.005;
  parameter Real d_f=0.9, g_fr=1.0;
  parameter Real priceFox=150, priceRabbit=12;
  input Real hunter(min=0, max=1, start=0.5);

initial equation
  der(foxes) = 20;
  value = 11000;

equation
  der(rabbits) =  rabbits*g_r - rabbits*foxes*d_rf - hunter*rabbits*0.01;
  der(foxes) = -foxes*d_f + rabbits*foxes*d_rf*g_fr - hunter*foxes*2;
  population = foxes+rabbits;
  value = priceFox*foxes + priceRabbit*rabbits;
end forest;

optimization forestOpt(objectiveIntegrand=-value)
  extends forest;
end forestOpt;
```

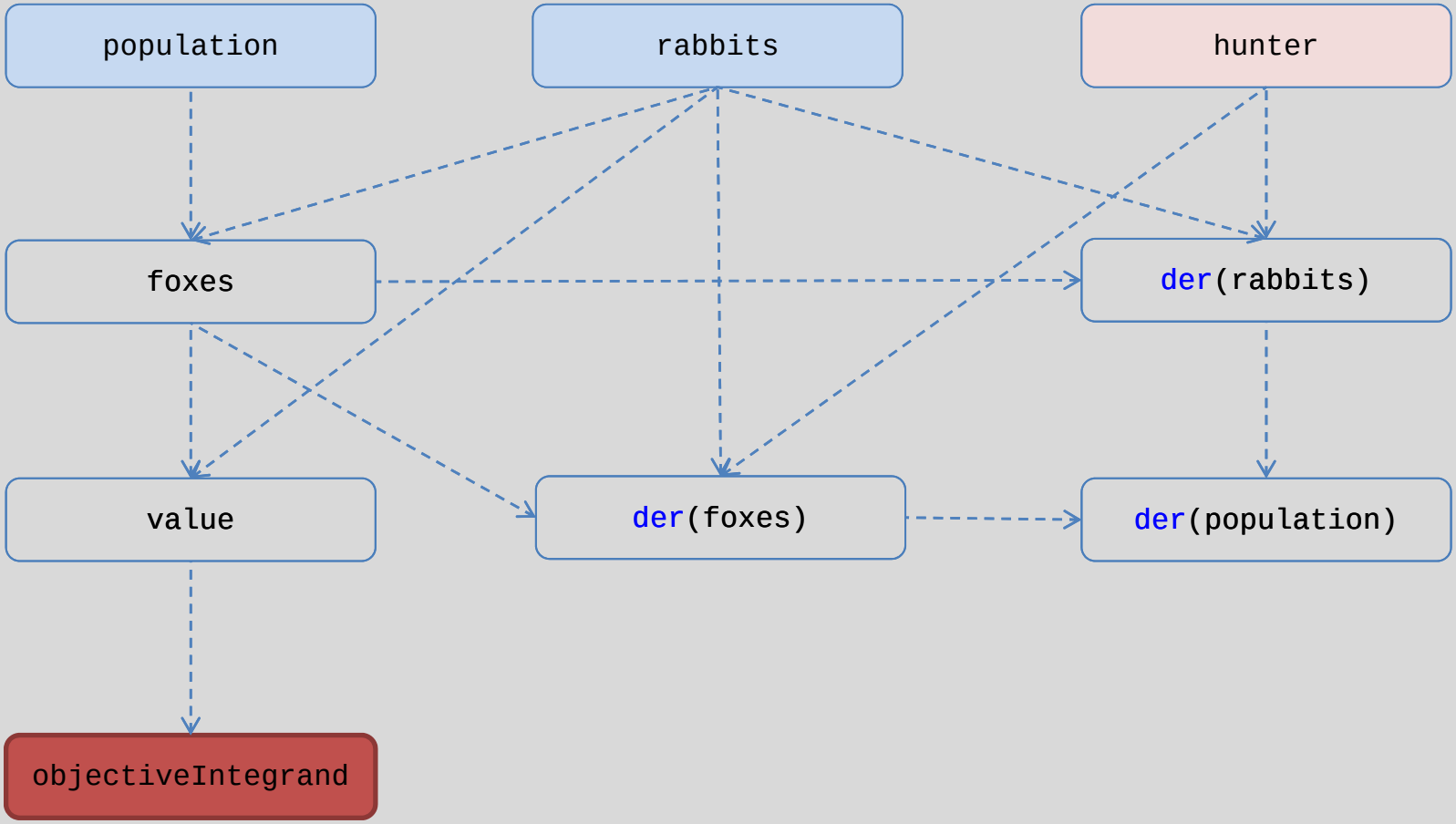
Example – Optimization Dependence graph



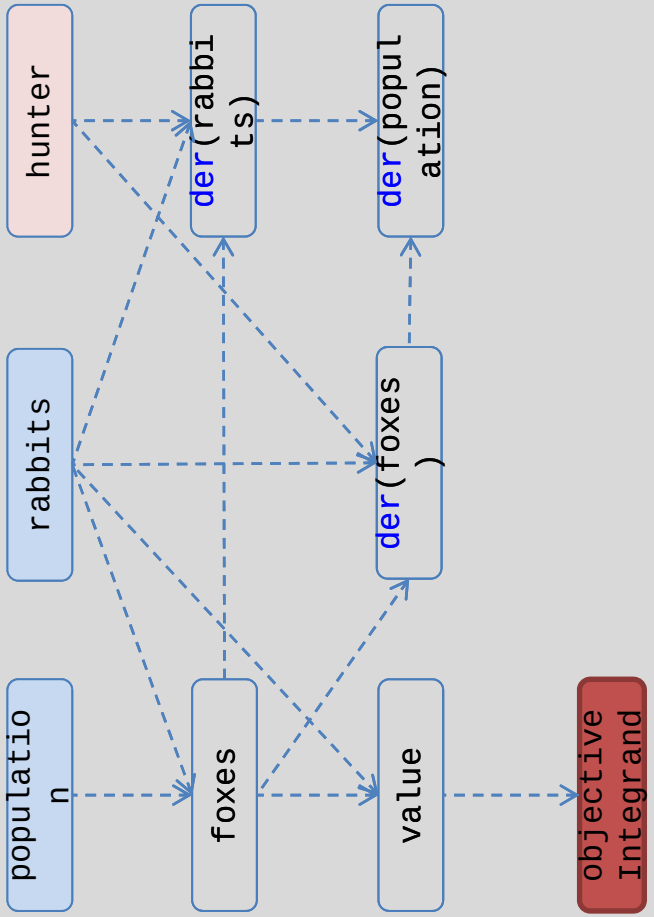
Example – Optimization Dependence graph



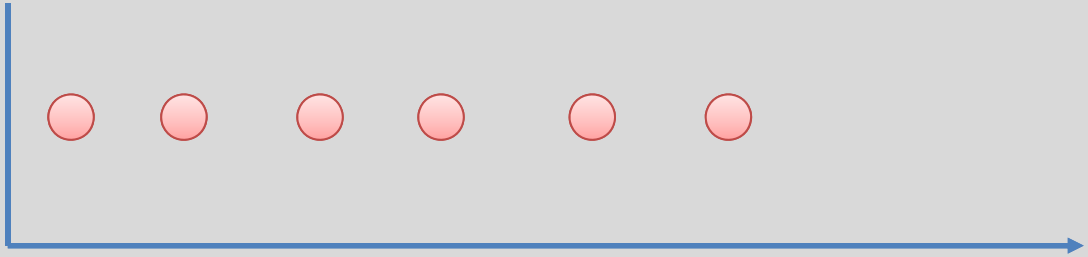
Example – Optimization Dependence graph

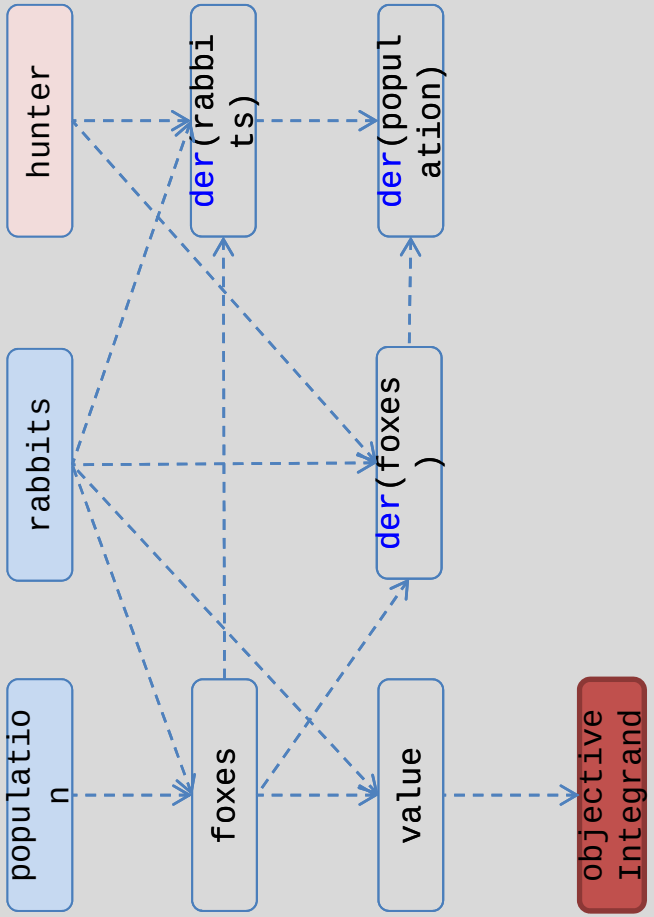


Example – Optimization horizon

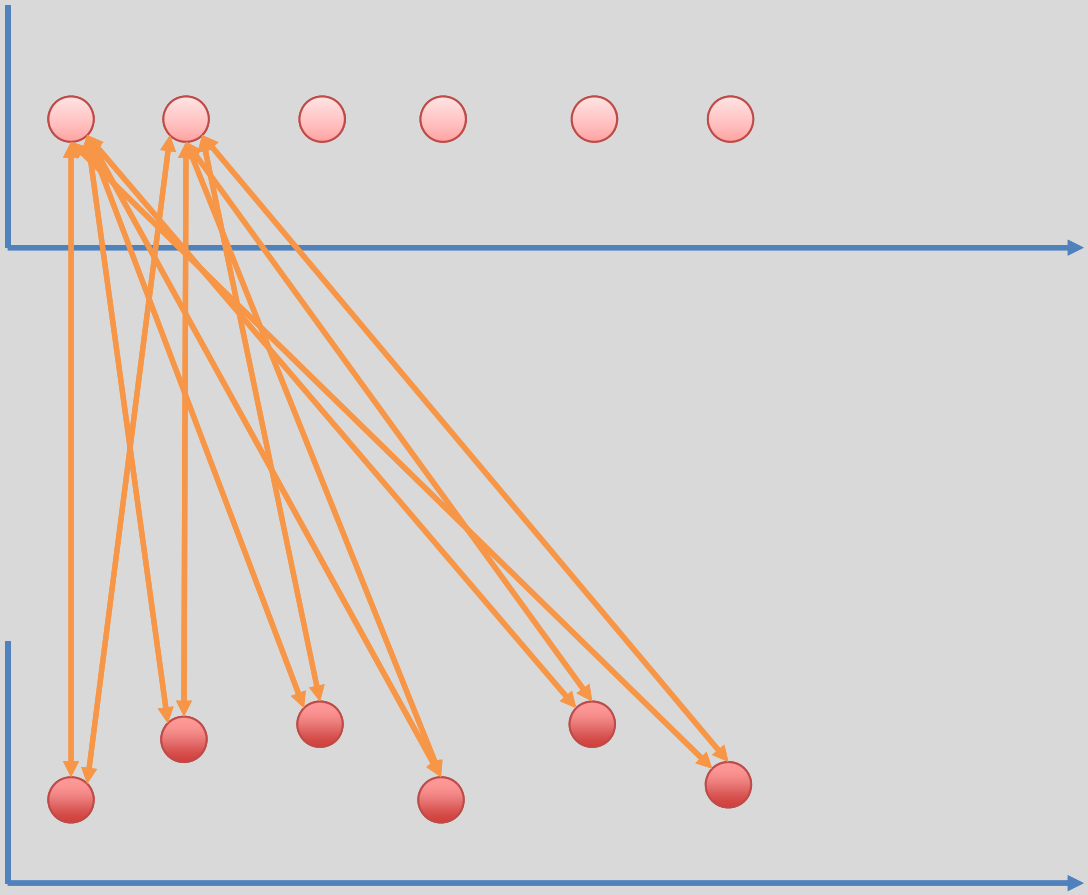


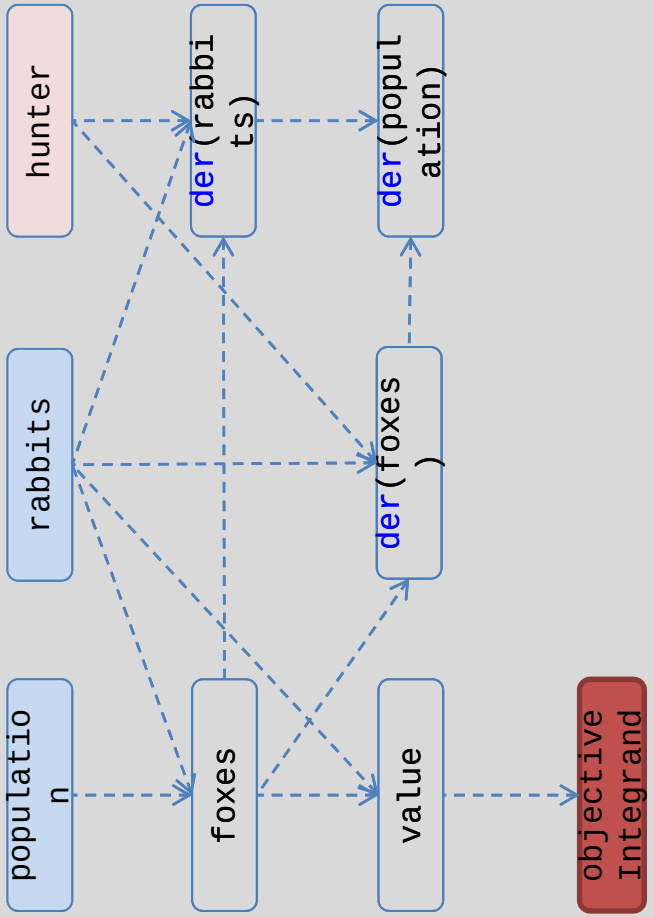
Example – Optimization horizon



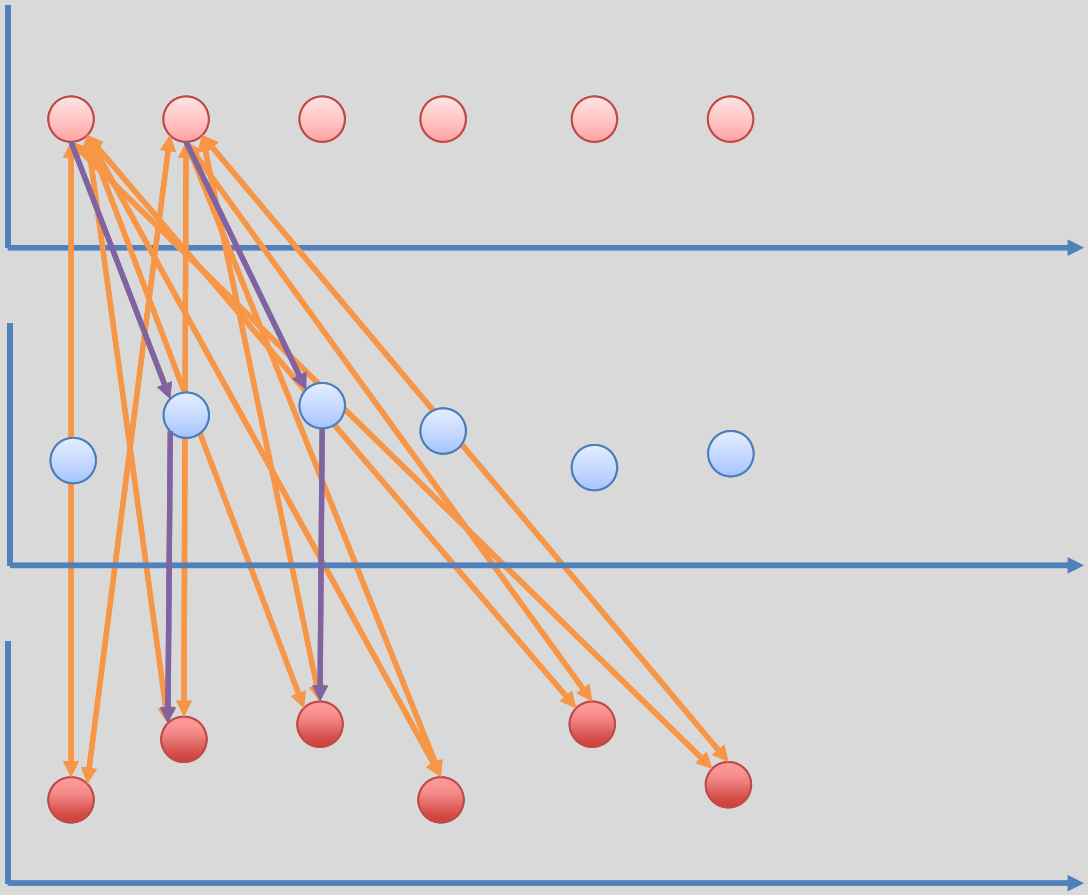


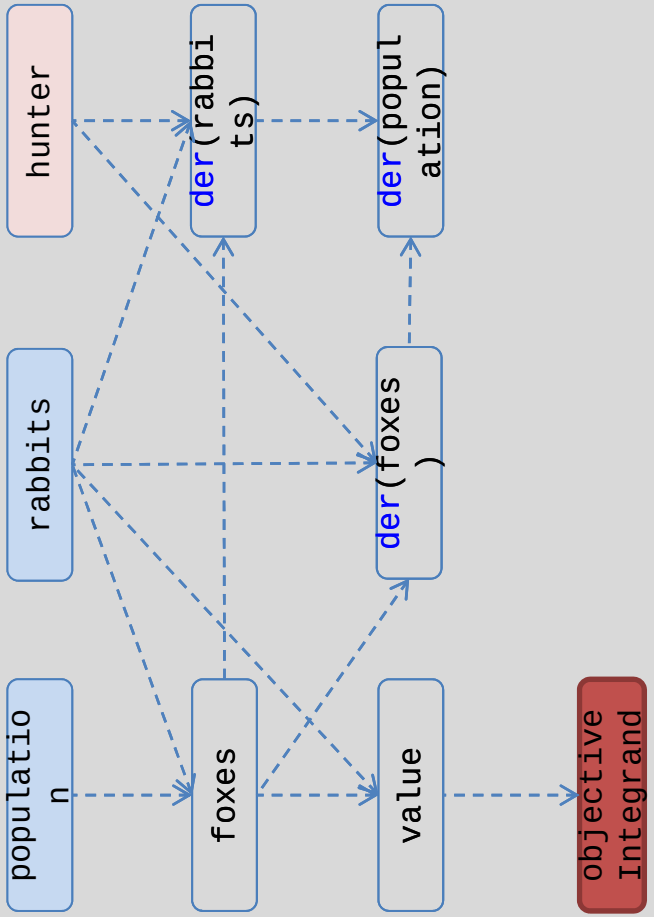
Example – Optimization horizon



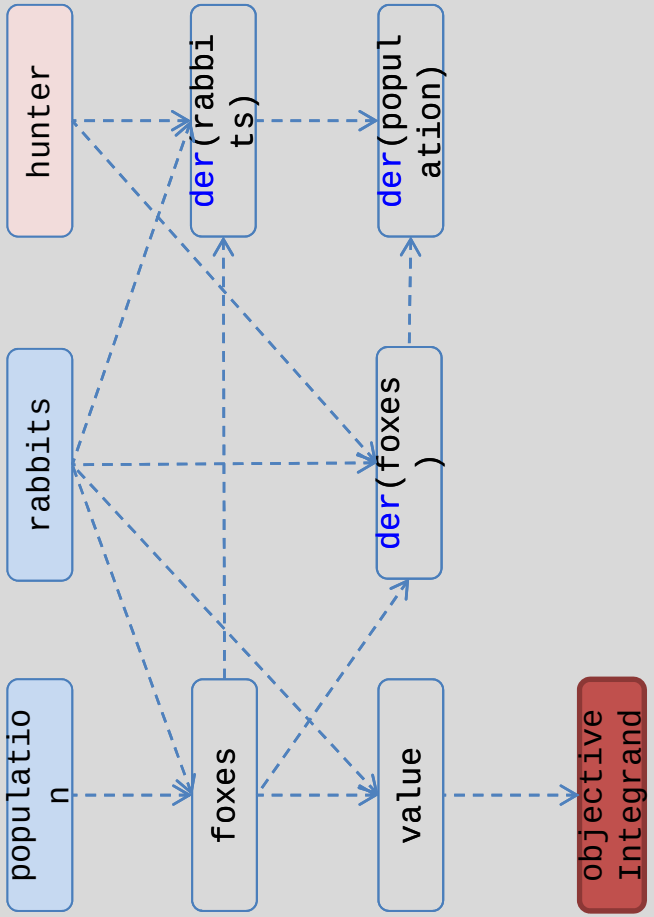


Example – Optimization horizon

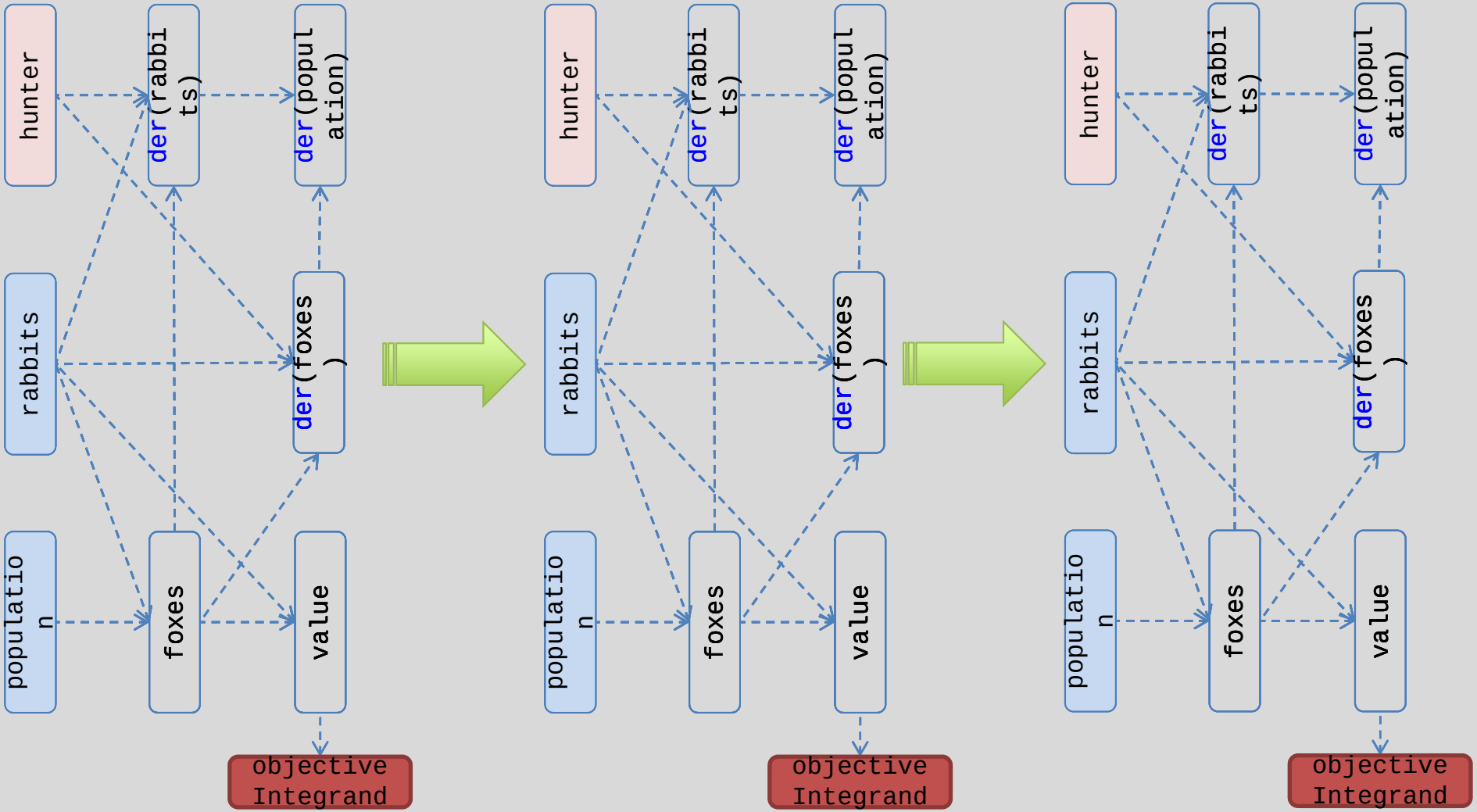




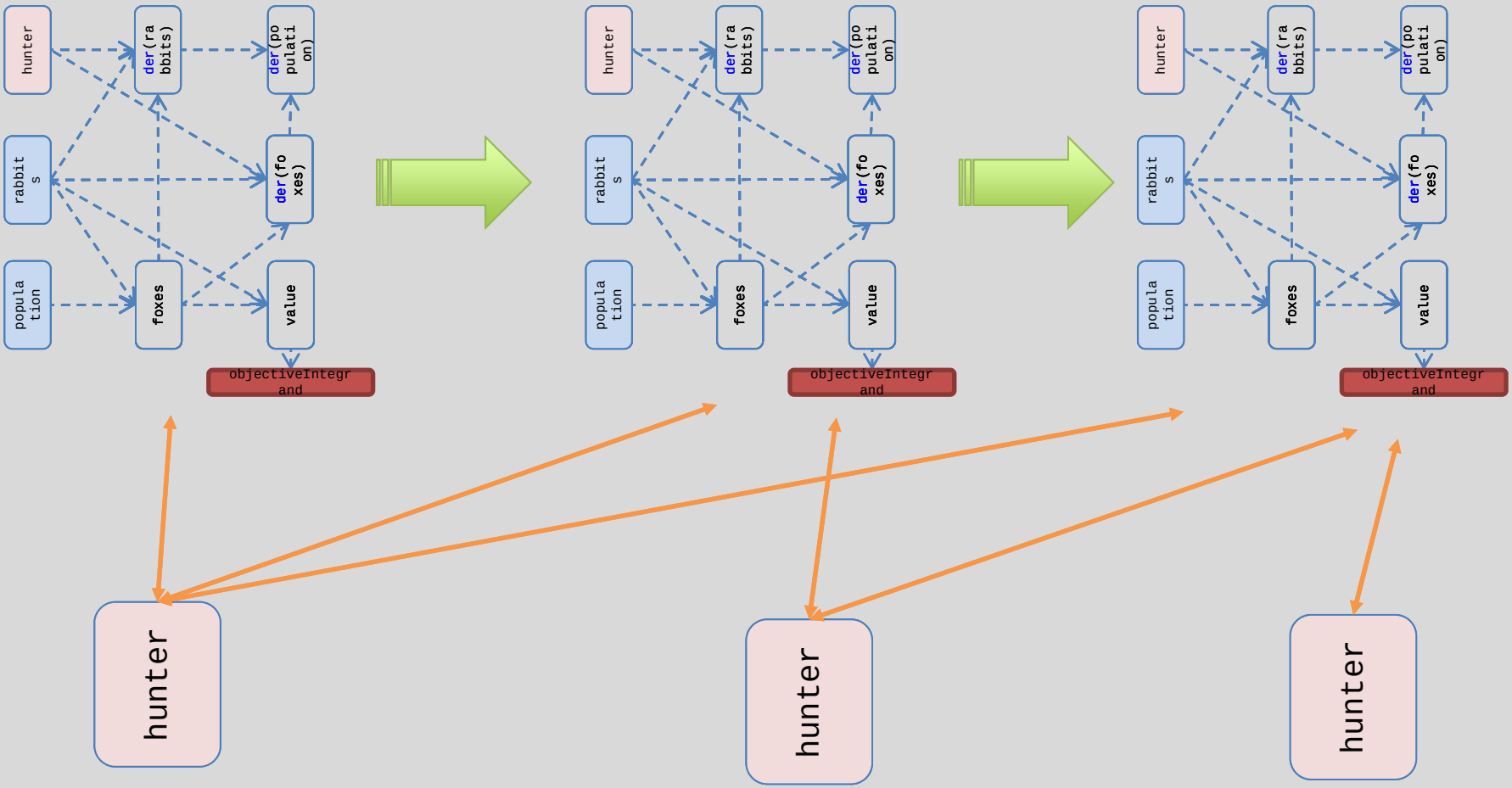
Example – Optimization horizon



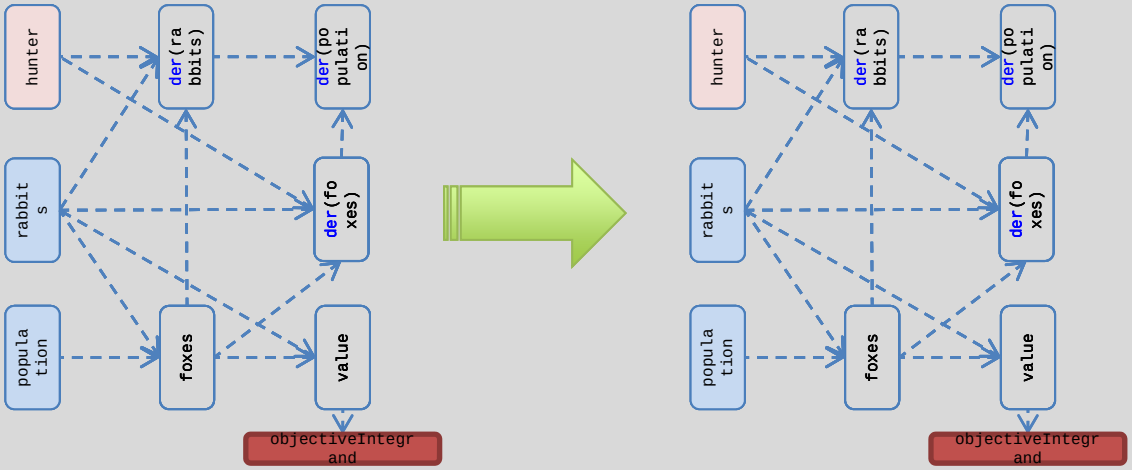
Example – Optimization horizon



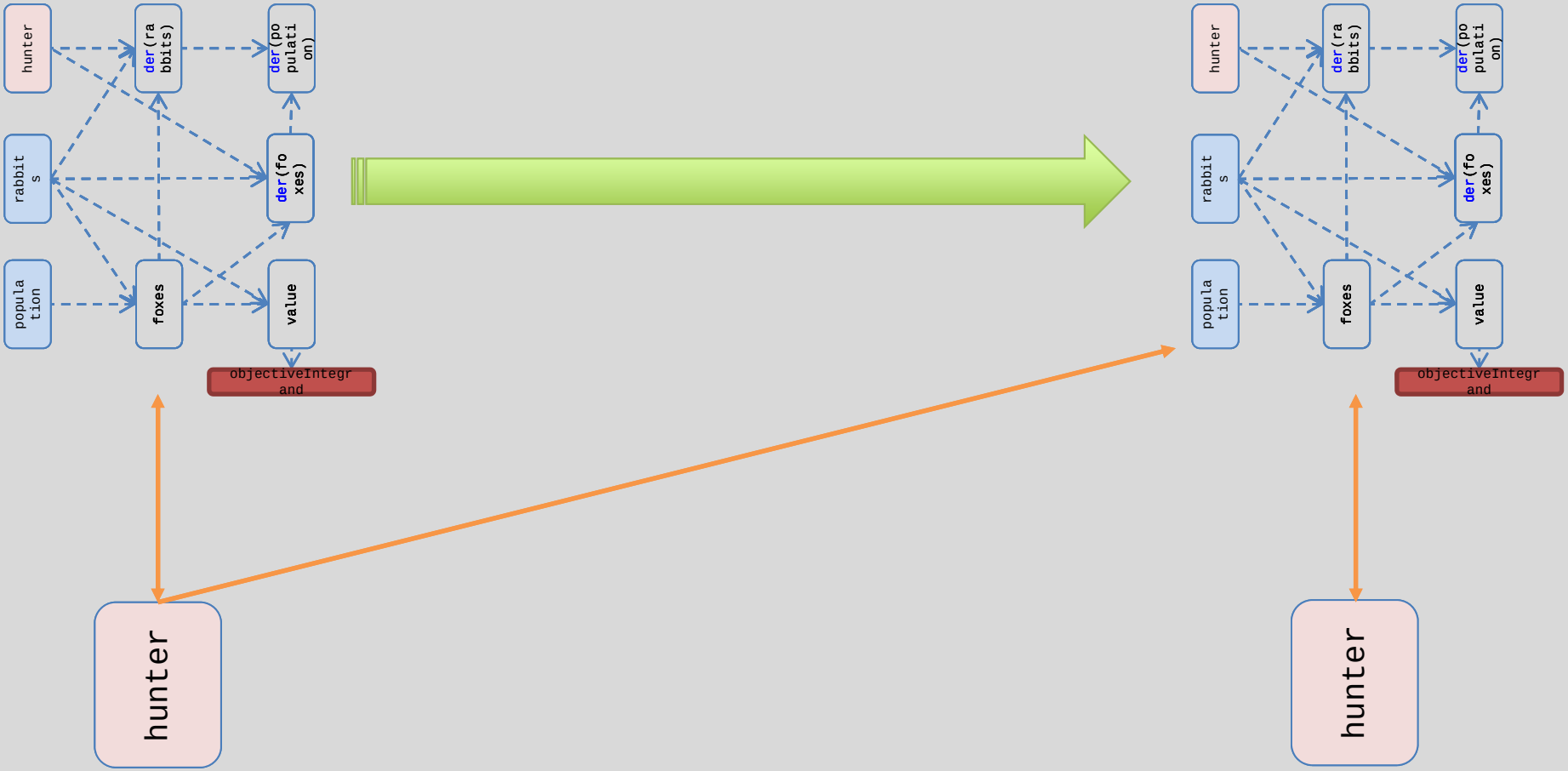
Example – Optimization split



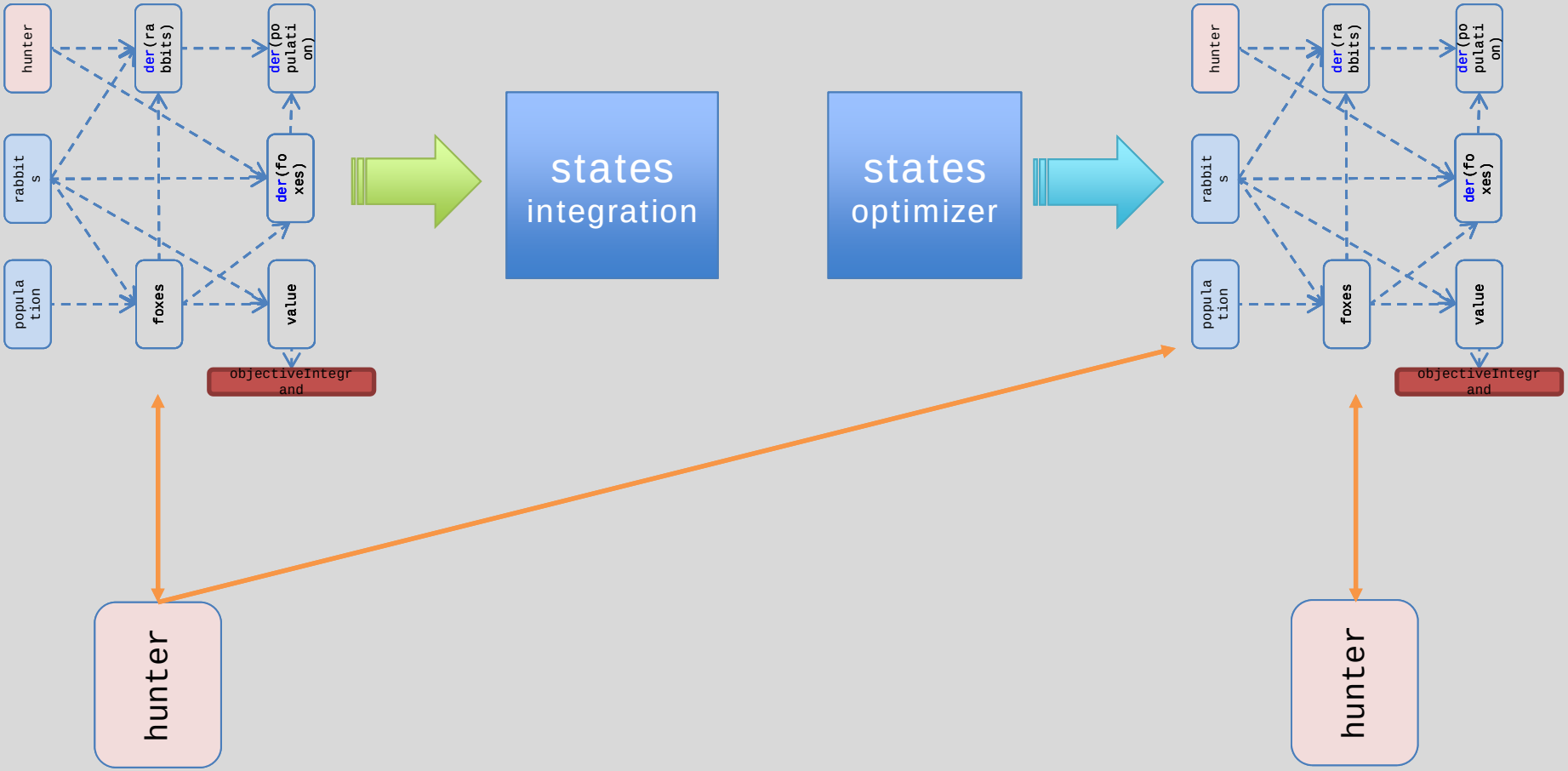
Example – Optimization split



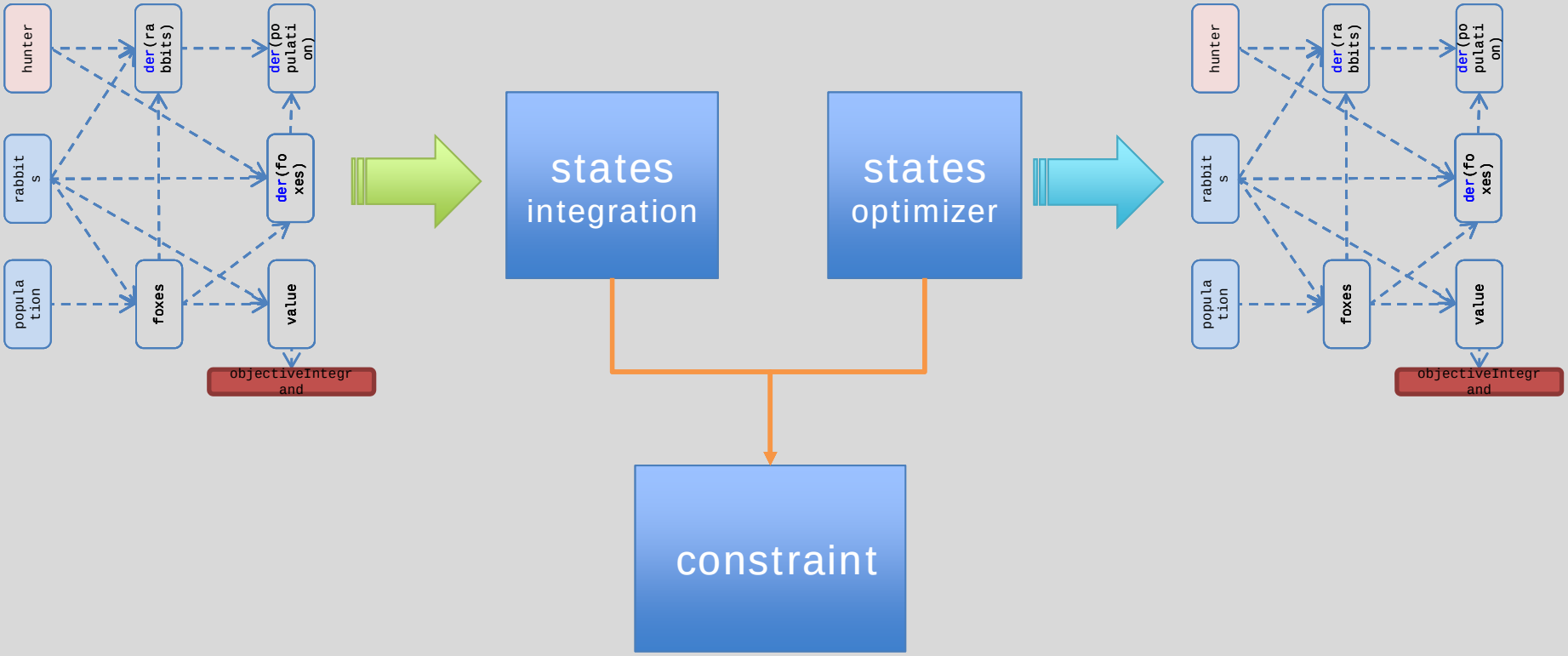
Example – Optimization split



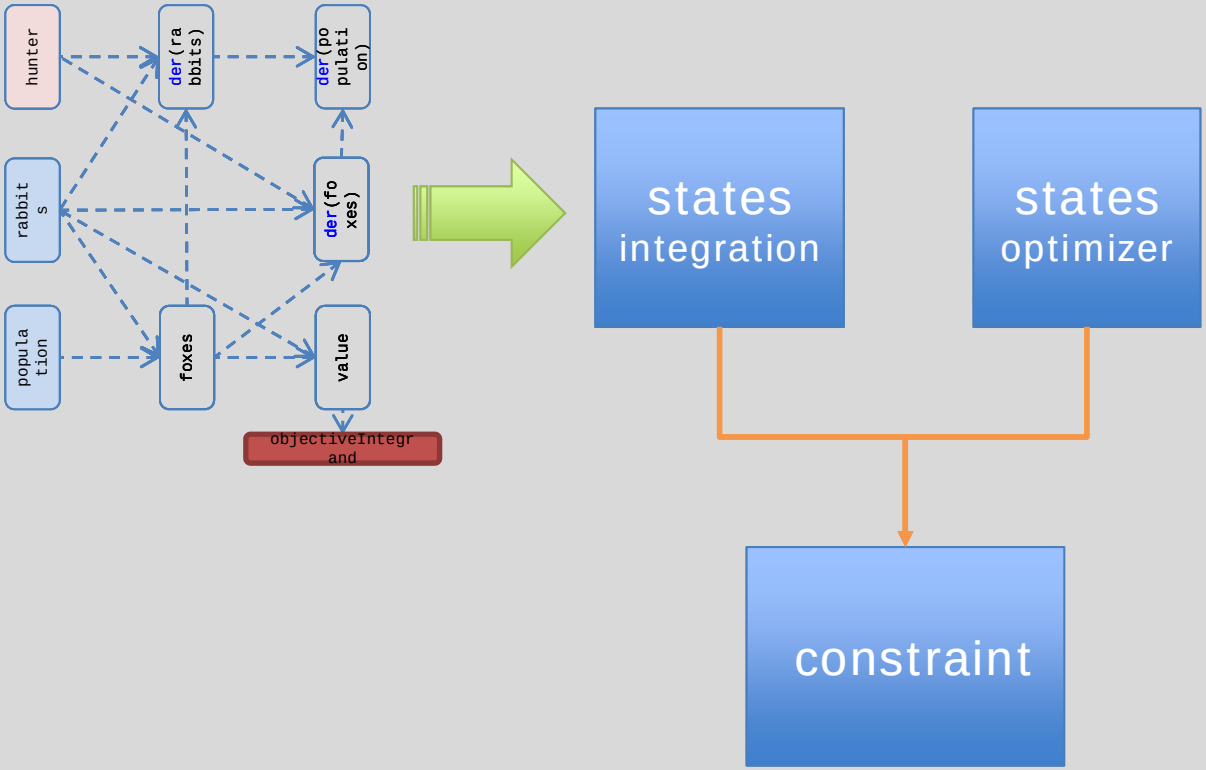
Example – Optimization split



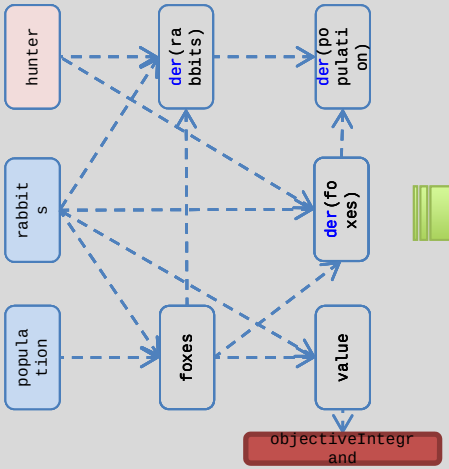
Example – Optimization split

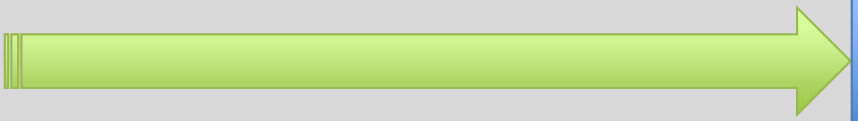


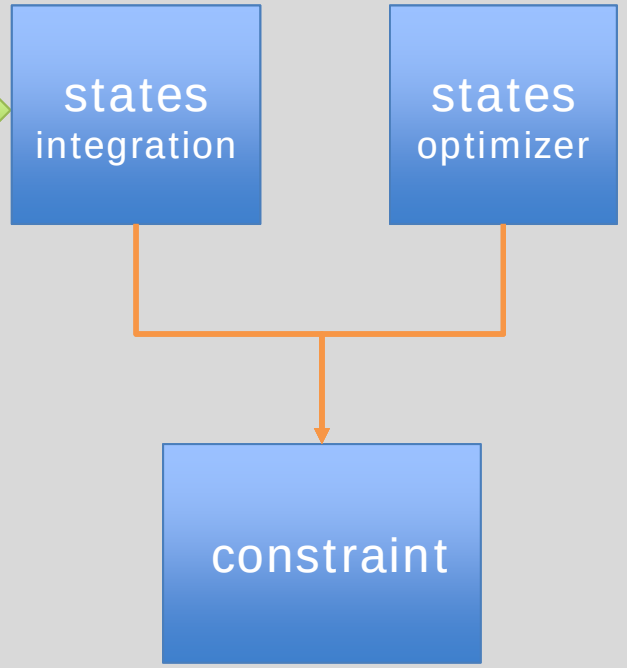
Example – Optimization split



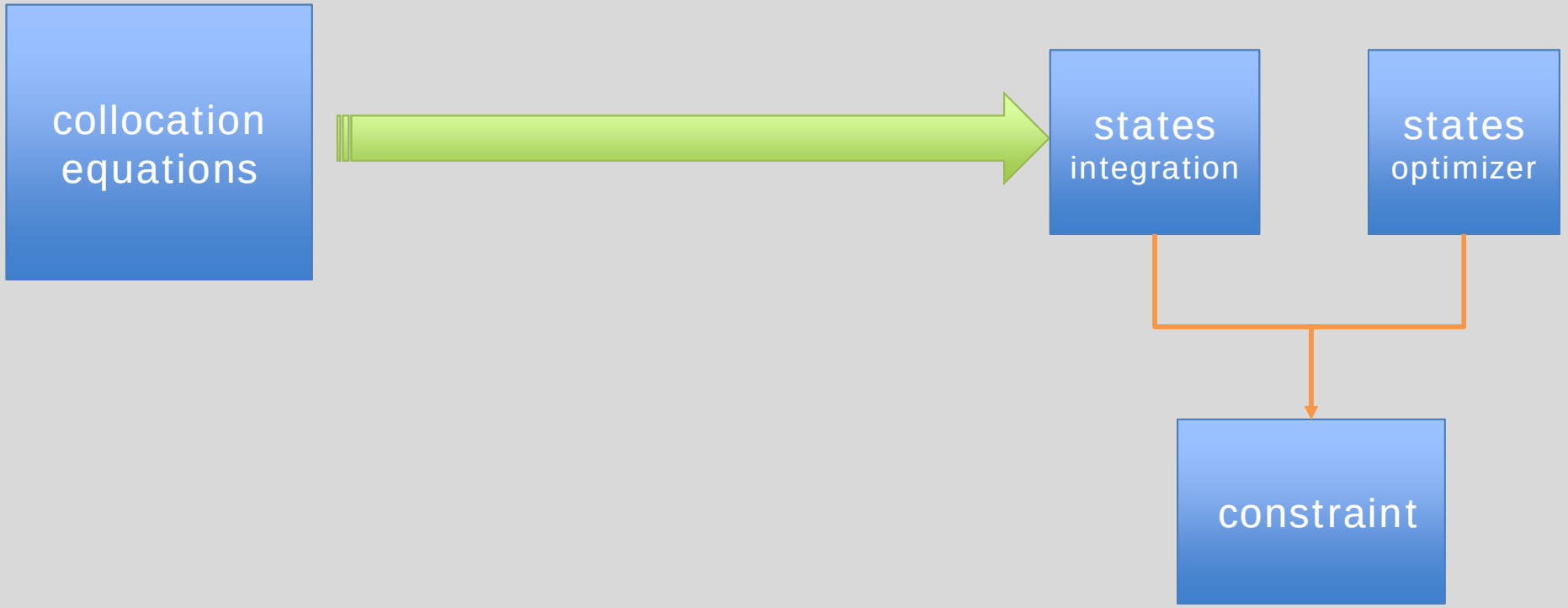
Example – Optimization split





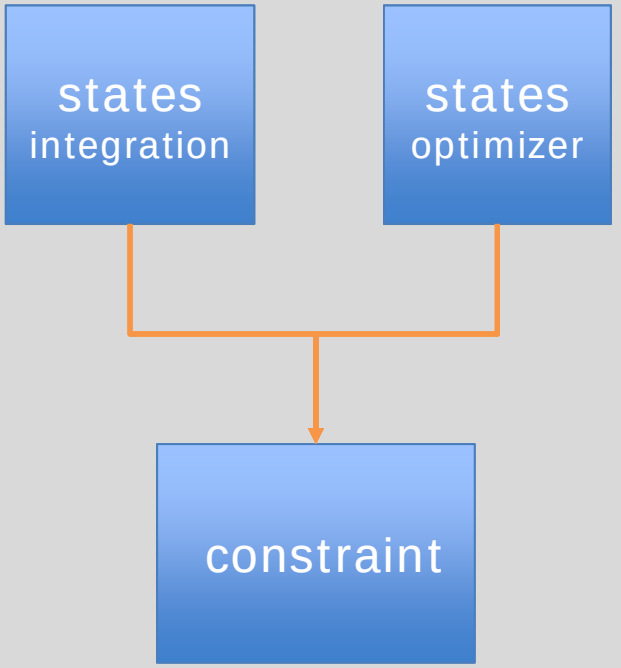


Example – Optimization split

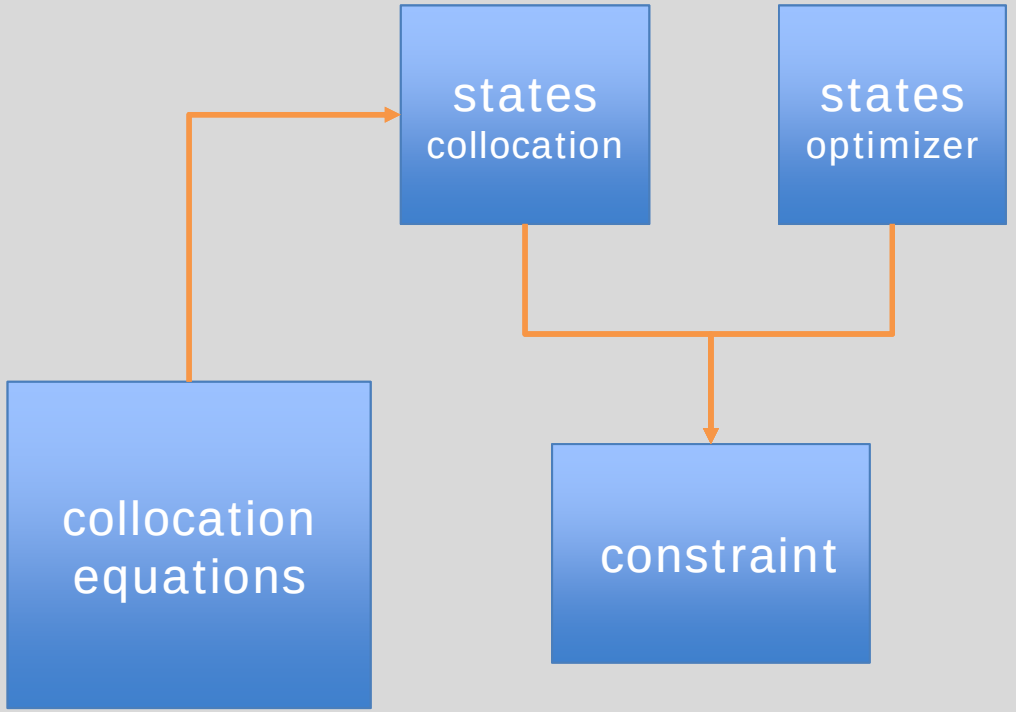


Example – Optimization split

collocation equations

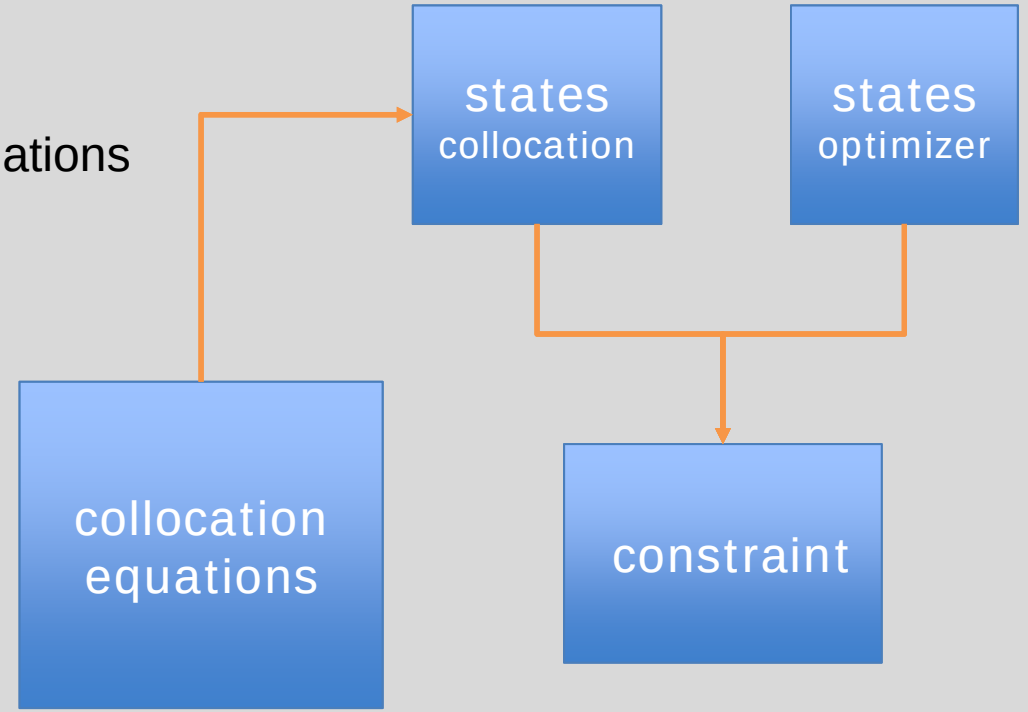


Example – Optimization split



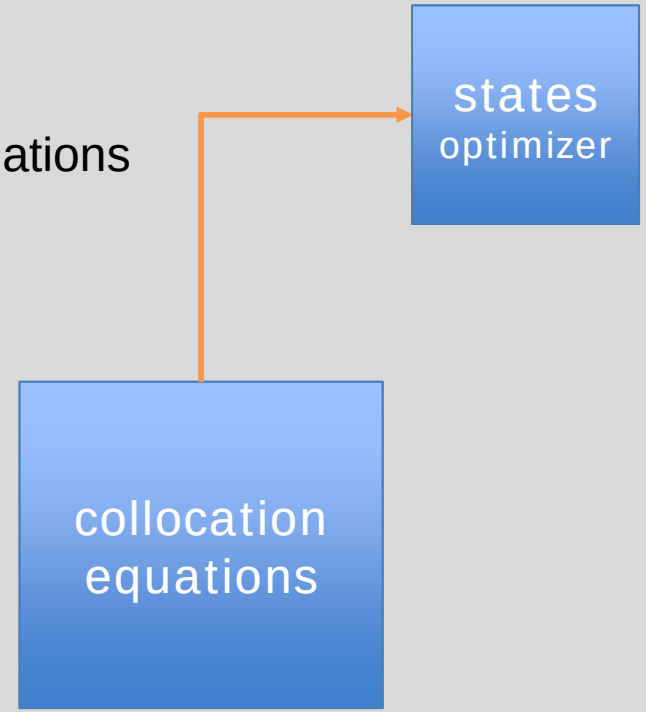
Example – Optimization split

Radau II A → constraints → Alias equations

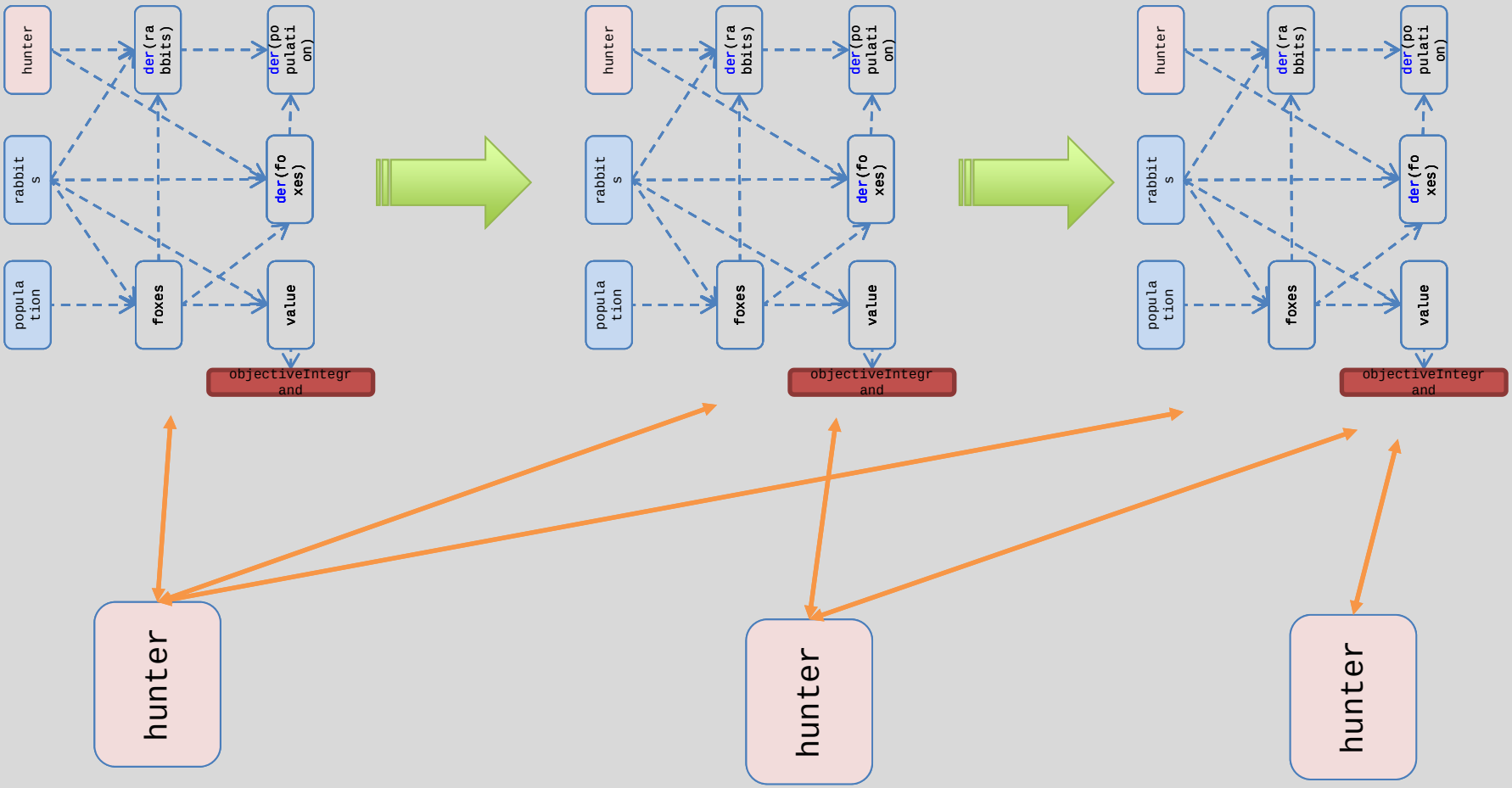


Example – Optimization split

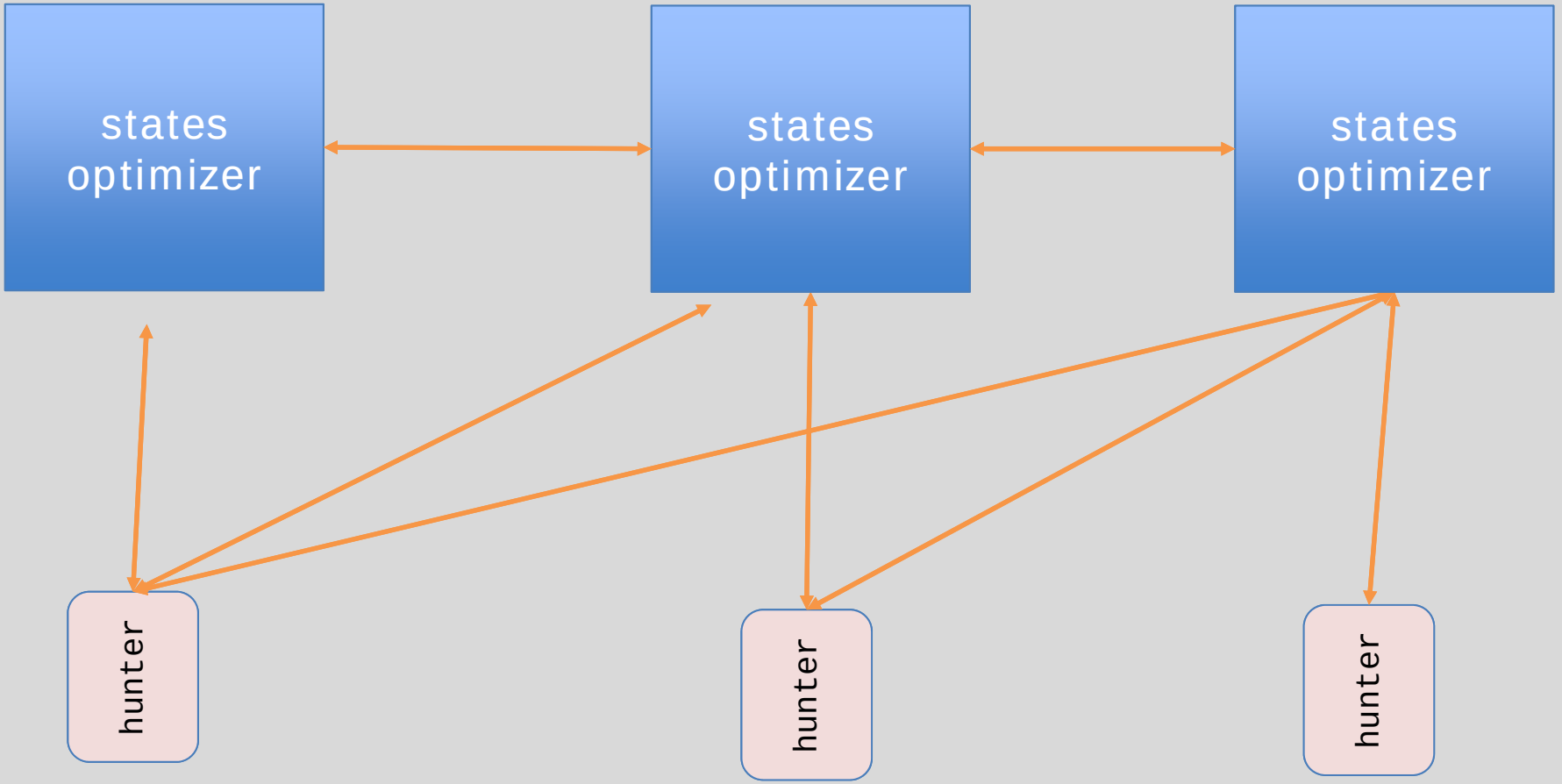
Radau II A → constraints → Alias equations



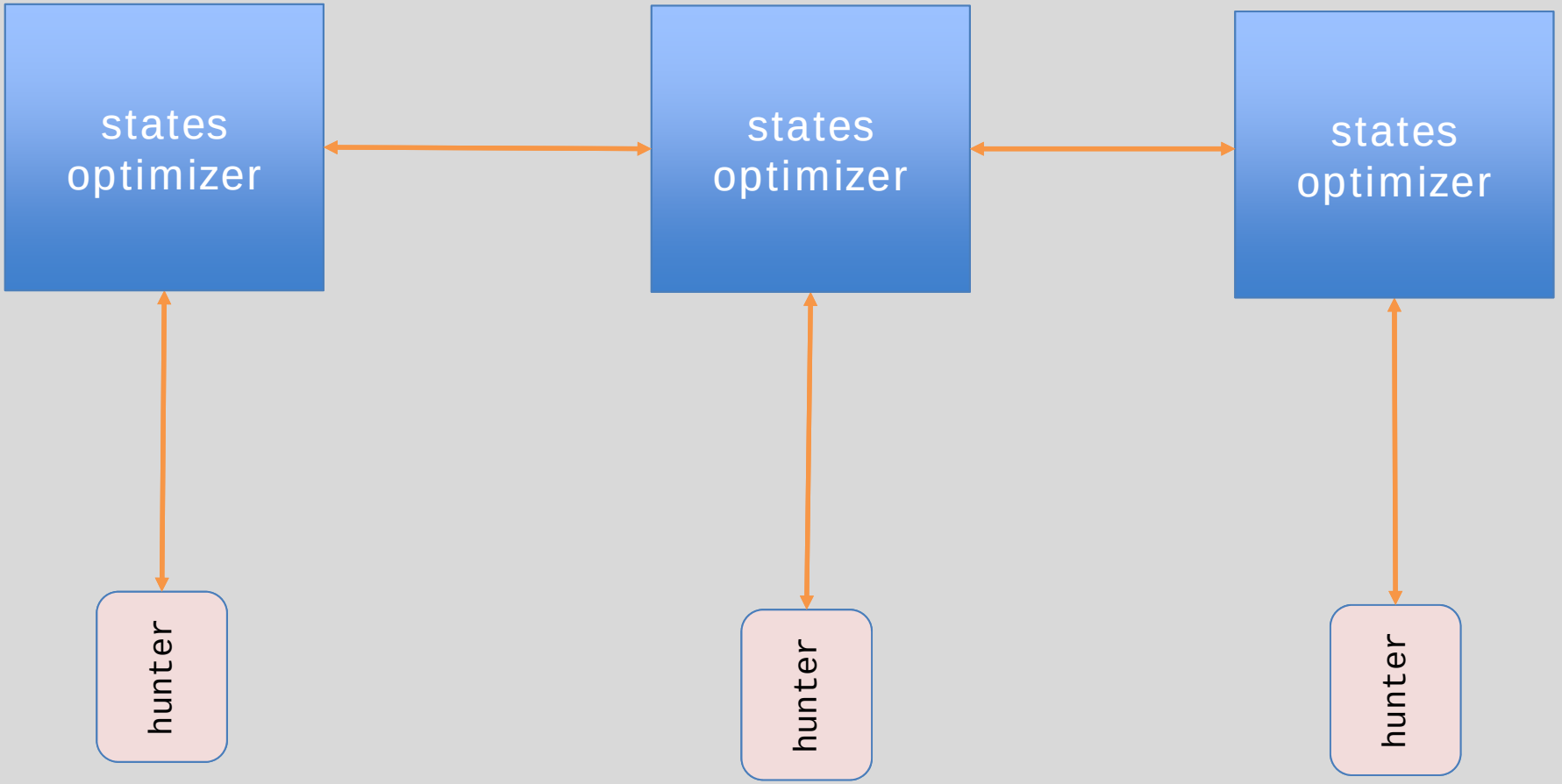
Example – Optimization split



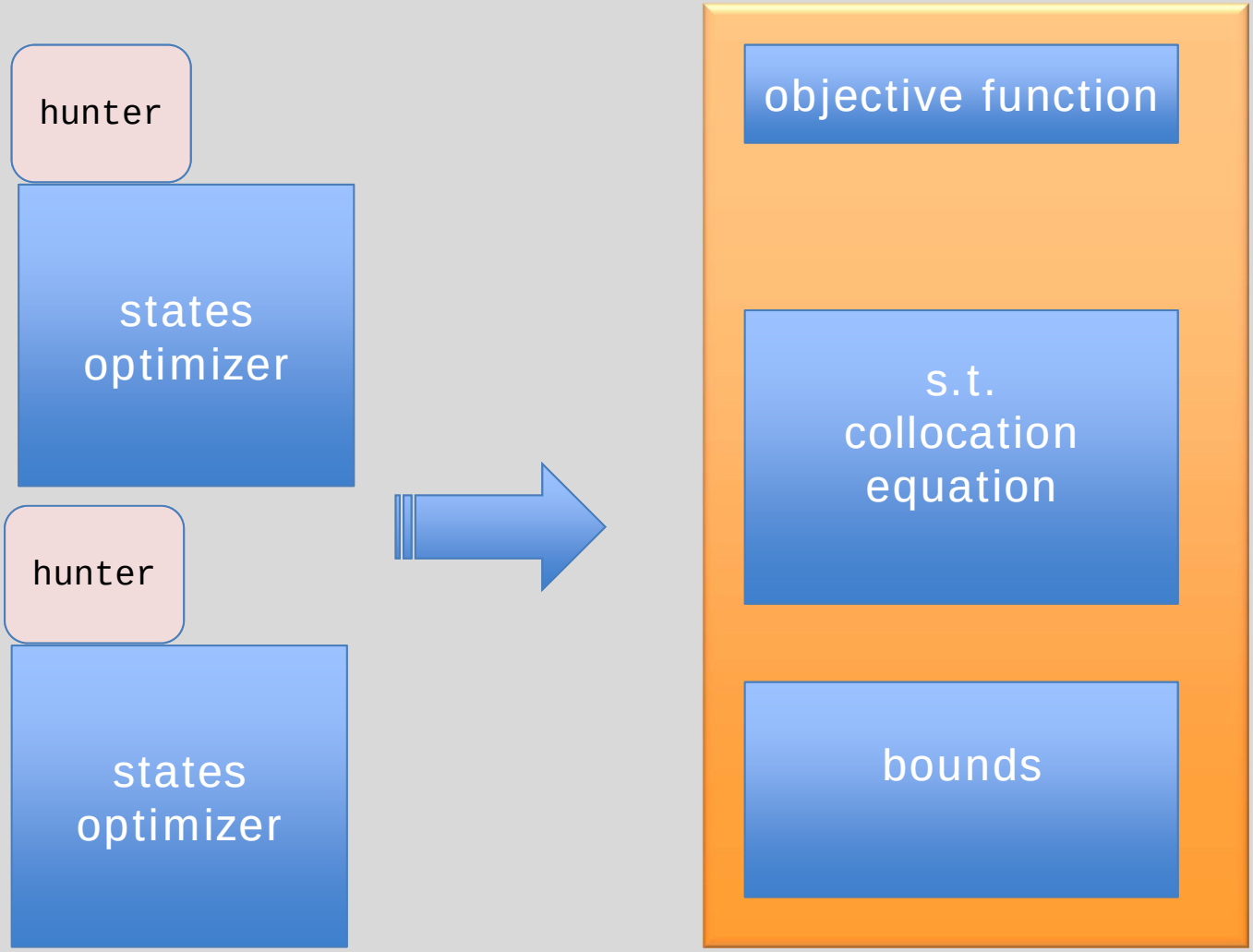
Example – Optimization split



Example – Optimization split

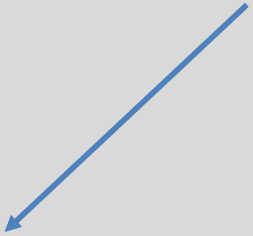


Example – Optimization optimizer

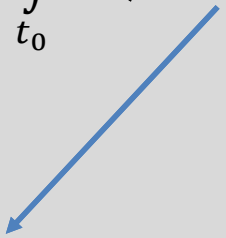


Extension Optimica

$$\min_{u(t)} J\left(\underline{x}(t), \underline{z}(t), \underline{p}, \underline{u}(t), t\right) = E\left(\underline{x}(t_f), \underline{z}(t_f), \underline{p}, \underline{u}(t_f), t_f\right) + \int_{t_0}^{t_f} L\left(\underline{x}(t), \underline{z}(t), \underline{p}, \underline{u}(t), t\right) \mathrm{dt}$$



objective



objectiveIntegrand

Extension Optimica

```
model forest
  Real foxes(min=0, max=100);
  Real rabbits(min=0, max=1000);
  Real population(start=850, min=550, max=1100, stateSelect=StateSelect.always);
  Real value;

  parameter Real g_r=0.4, d_rf=0.005;
  parameter Real d_f=0.9, g_fr=1.0;
  parameter Real priceFox=150, priceRabbit=12;
  input Real hunter(min=0, max=1, start=0.5);

initial equation
  der(foxes) = 20;
  value = 11000;

equation
  der(rabbits) =  rabbits*g_r - rabbits*foxes*d_rf - hunter*rabbits*0.01;
  der(foxes) = -foxes*d_f + rabbits*foxes*d_rf*g_fr - hunter*foxes*2;
  population = foxes+rabbits;
  value = priceFox*foxes + priceRabbit*rabbits;
end forest;

optimization forestOpt(objectiveIntegrand=-value)
  extends forest;
end forestOpt;
```

Extension Optimica

```
model forest
  Real foxes(min=0, max=100);
  Real rabbits(min=0, max=1000);
  Real population(start=850, min=550, max=1100, stateSelect=StateSelect.always);
  Real value;

  parameter Real g_r=0.4, d_rf=0.005;
  parameter Real d_f=0.9, g_fr=1.0;
  parameter Real priceFox=150, priceRabbit=12;
  input Real hunter(min=0, max=1, start=0.5);

initial equation
  der(foxes) = 20;
  value = 11000;

equation
  der(rabbits) =  rabbits*g_r - rabbits*foxes*d_rf - hunter*rabbits*0.01;
  der(foxes) = -foxes*d_f + rabbits*foxes*d_rf*g_fr - hunter*foxes*2;
  population = foxes+rabbits;
  value = priceFox*foxes + priceRabbit*rabbits;
end forest;
```

```
optimization forestOpt(objectiveIntegrand=-value)
  extends forest;
end forestOpt;
```

Extension Optimica

- Not supported
  - Time point operator
    - x(0.5), cost(finalTime)
  - ***free*** attribute
    - using for parameter
- limited
  - constraint-block