Bridging the Gap Between Clone-and-Own and Software Product Lines

Timo
Kehrer[1]

Thomas
Thüm[2]

Alexander
Schultheiß[1]

Paul Maximilian
Bittner[2]

HUMBOLDT-UNIVERSITÄT ZU BERLIN [1]

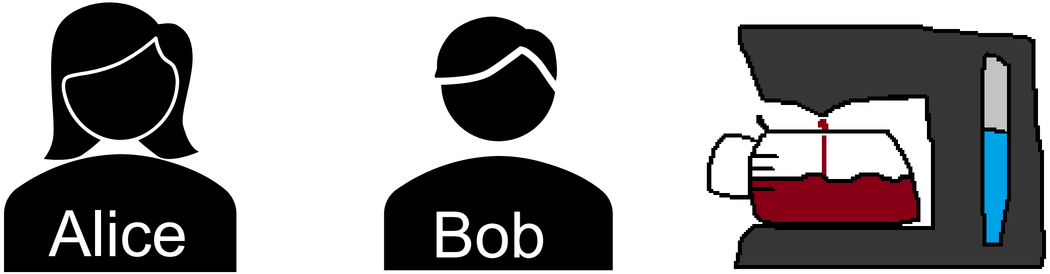
[2] universität ulm

Software Engineering in the Past

Hey Bob, did you know you should compare `strings` using `equals`? I just fixed my code!

Incredible! I need that fix!

But I also changed a lot of other stuff! ☹

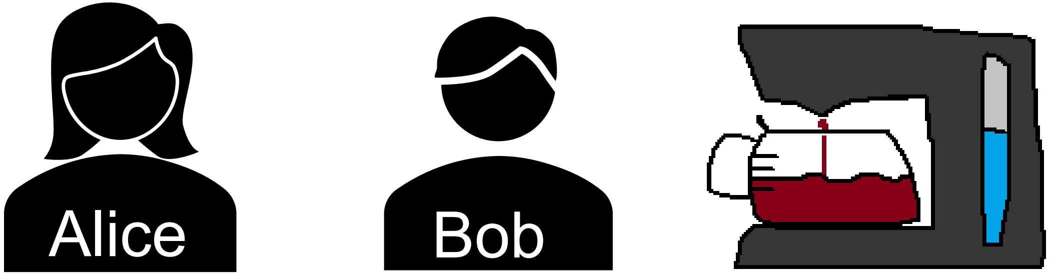


Software Engineering in the Future

Hey Bob, did you know you should compare `strings` using `equals`? I just fixed my code!

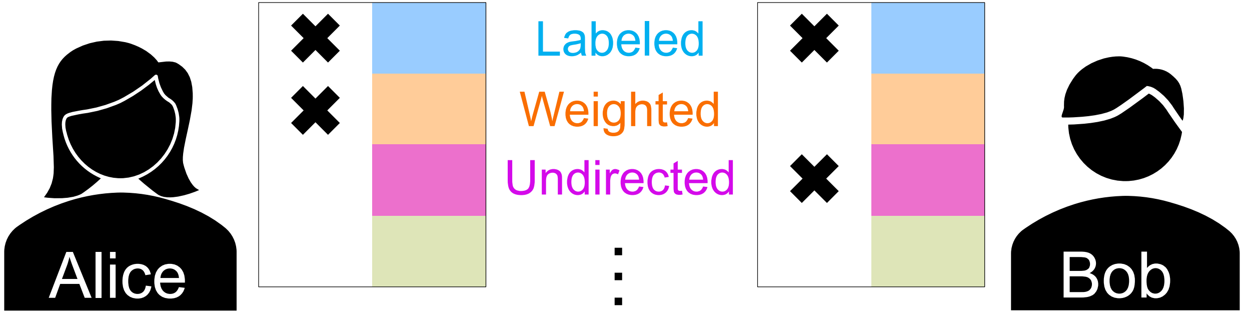
Incredible! I need that fix!

Merging is easy because of **VariantSync**!



```
boolean equals(Edge e) {
  return weight == e.weight
    && label == e.label
    && source == e.source
    && target == e.target;
}
```

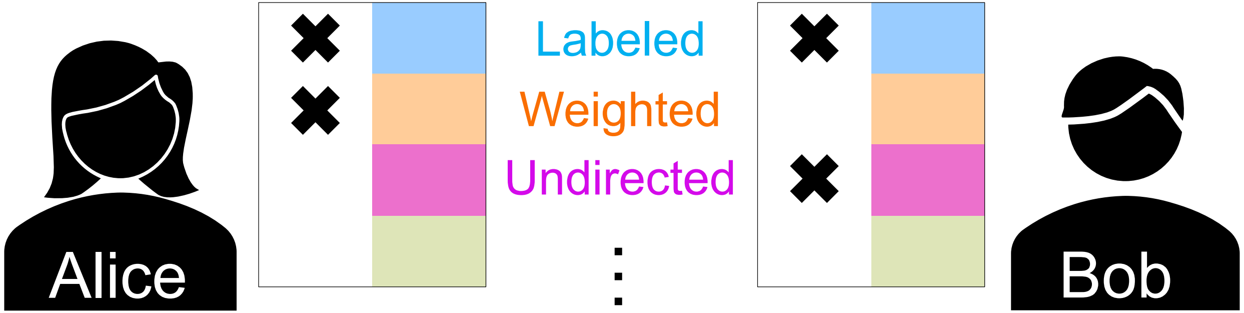
```
boolean equals(Edge e) {
  return label == e.label
    && (source == e.source && target == e.target
      || source == e.target && target == e.source);
}
```



Alice		Bob
✖	Labeled	✖
✖	Weighted	
	Undirected	✖
	⋮	

```
boolean equals(Edge e) {
  return weight == e.weight
    && label.equals(e.label)
    && source == e.source
    && target == e.target;
}
```

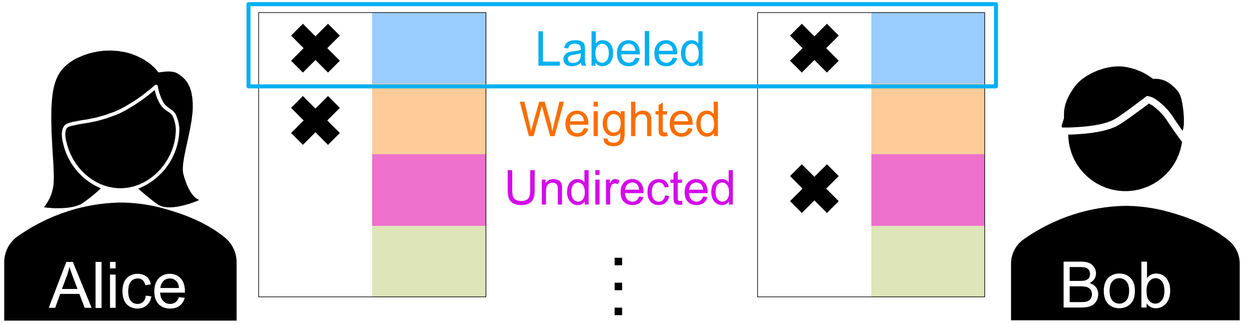
```
boolean equals(Edge e) {
  return label == e.label
    && (source == e.source && target == e.target
      || source == e.target && target == e.source);
}
```



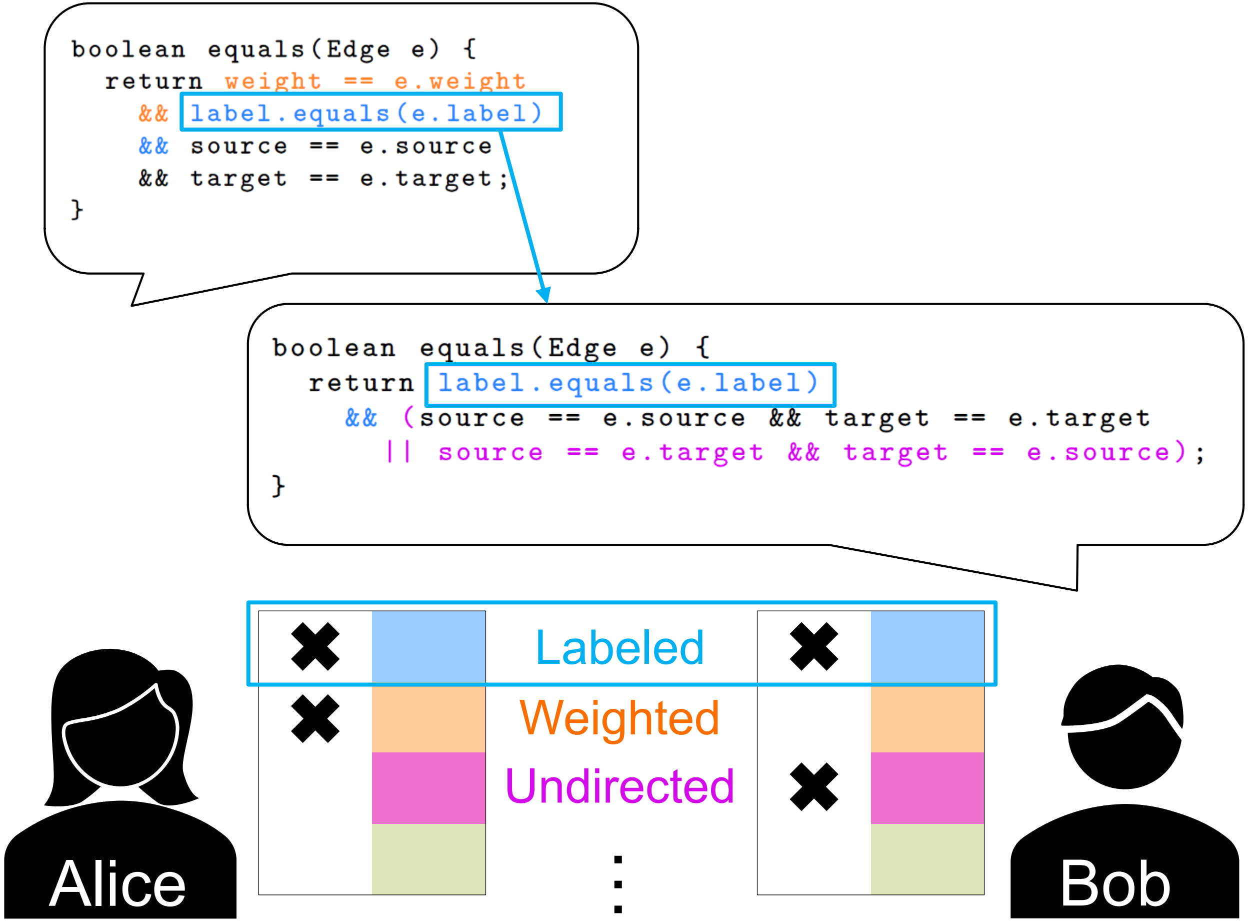
Labeled
Weighted
Undirected
⋮

```
boolean equals(Edge e) {
  return weight == e.weight
    && label.equals(e.label)
    && source == e.source
    && target == e.target;
}
```

```
boolean equals(Edge e) {
  return label == e.label
    && (source == e.source && target == e.target
      || source == e.target && target == e.source);
}
```



Alice		Bob
✖	Labeled	✖
✖	Weighted	
	Undirected	✖
	⋮	



```
boolean equals(Edge e) {
  return weight == e.weight
    && label.equals(e.label)
    && source == e.source
    && target == e.target;
}
```

```
boolean equals(Edge e) {
  return label.equals(e.label)
    && (source == e.source && target == e.target
      || source == e.target && target == e.source);
}
```

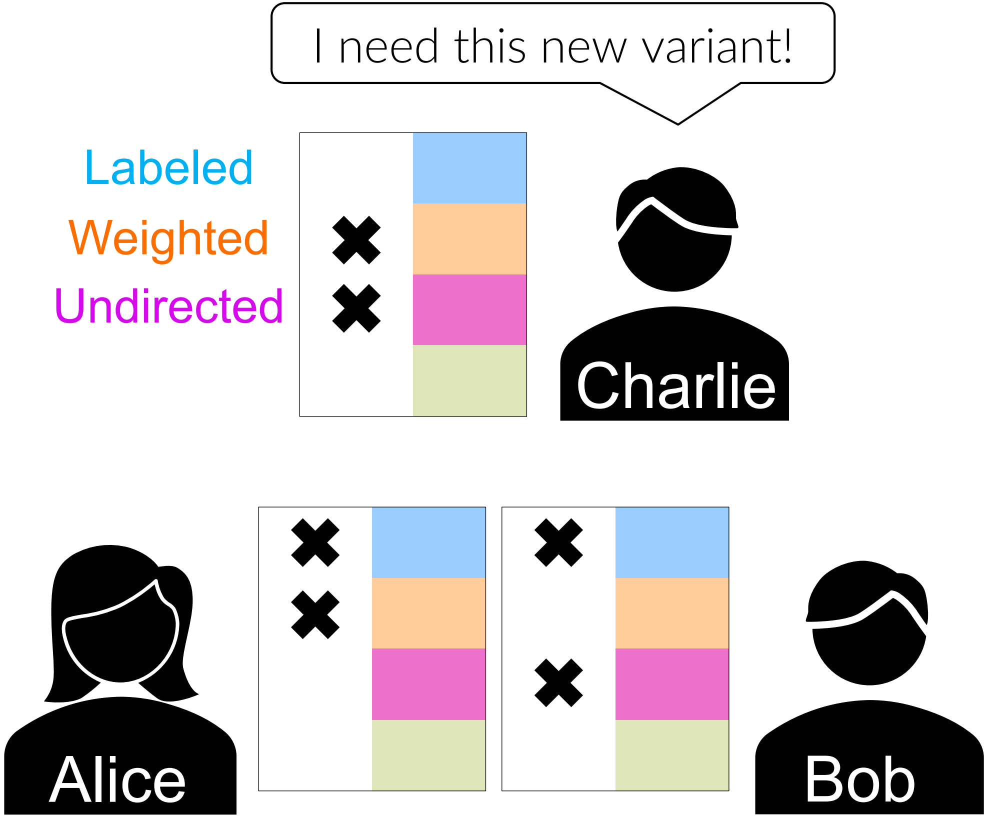
Alice			Bob	
✖	🟦	Labeled	✖	🟦
✖	🟧	Weighted		🟧
	🟪	Undirected	✖	🟪
	🟩	⋮		🟩

What Did Alice Do?

```
boolean equals(Edge e) {
  return label == e.label
    && source == e.source
    && target == e.target;
}
```

```
boolean equals(Edge e) {
  return weight == e.weight
    && label == e.label
    && source == e.source
    && target == e.target;
}
```

Can you imagine, we have a new customer!



I need this new variant!

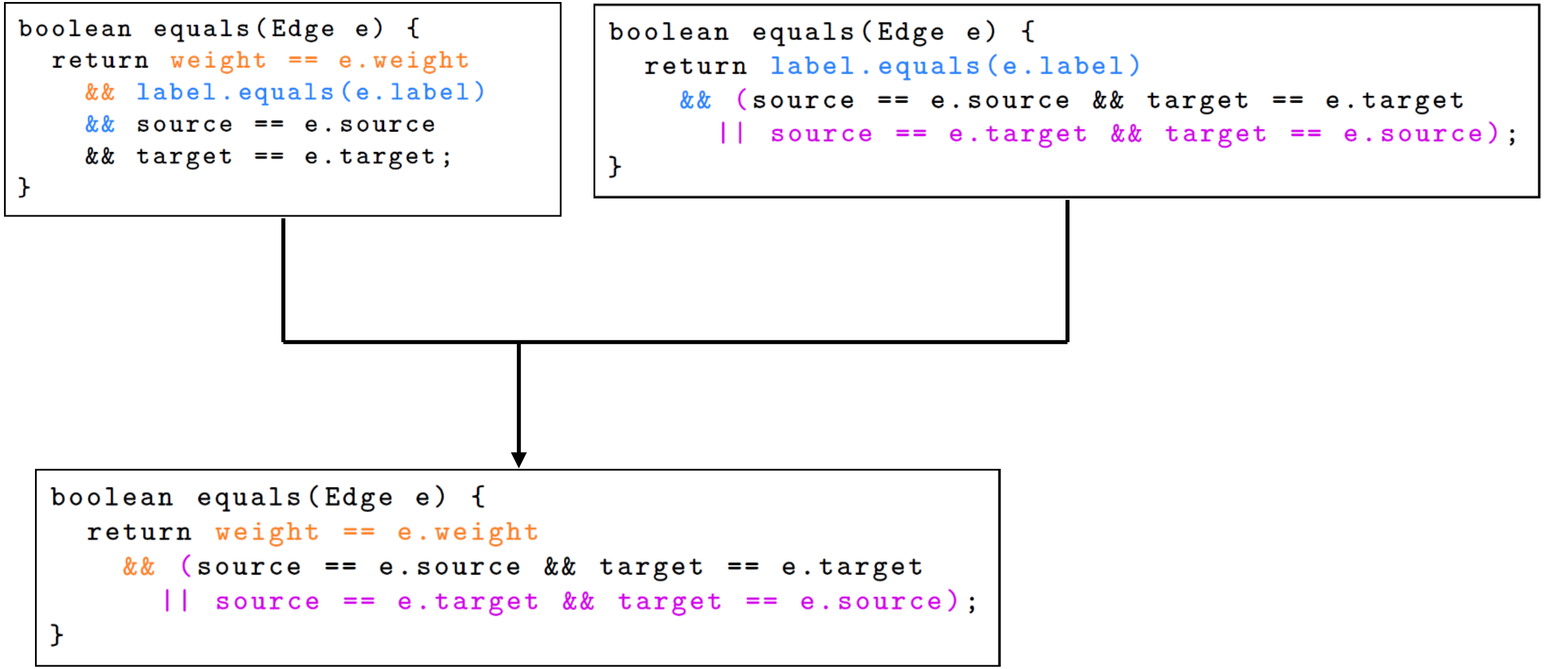
Labeled

Weighted

Undirected

Alice‘s variant is a possible donor for Weighted

Bob‘s variant is a possible donor for Undirected

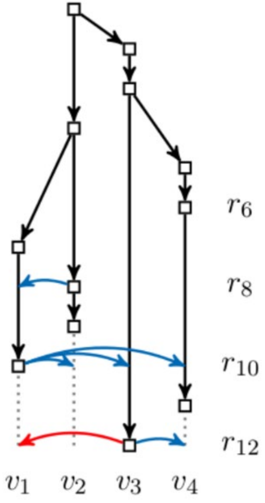


```
boolean equals(Edge e) {
  return weight == e.weight
    && label.equals(e.label)
    && source == e.source
    && target == e.target;
}
```

```
boolean equals(Edge e) {
  return label.equals(e.label)
    && (source == e.source && target == e.target
      || source == e.target && target == e.source);
}
```

```
boolean equals(Edge e) {
  return weight == e.weight
    && (source == e.source && target == e.target
      || source == e.target && target == e.source);
}
```

Evolution of variants



Feature context

Labeled

¬Weighted

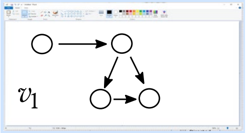
Graph

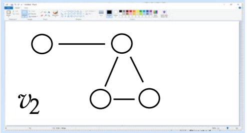
¬ Undirected ∨ Weighted

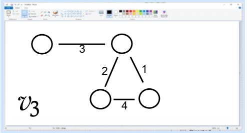
```
boolean equals(Edge e) {
  return weight == e.weight
    && label.equals(e.label)
    && source == e.source
    && target == e.target;
}
```

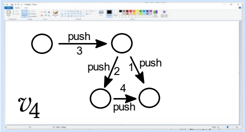
Feature mappings

Variants









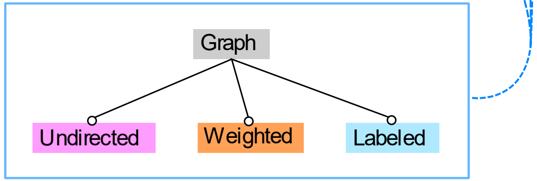
Configurations

■ Graph
- □ Undirected
- □ Weighted
- □ Labeled

■ Graph
- ☑ Undirected
- □ Weighted
- □ Labeled

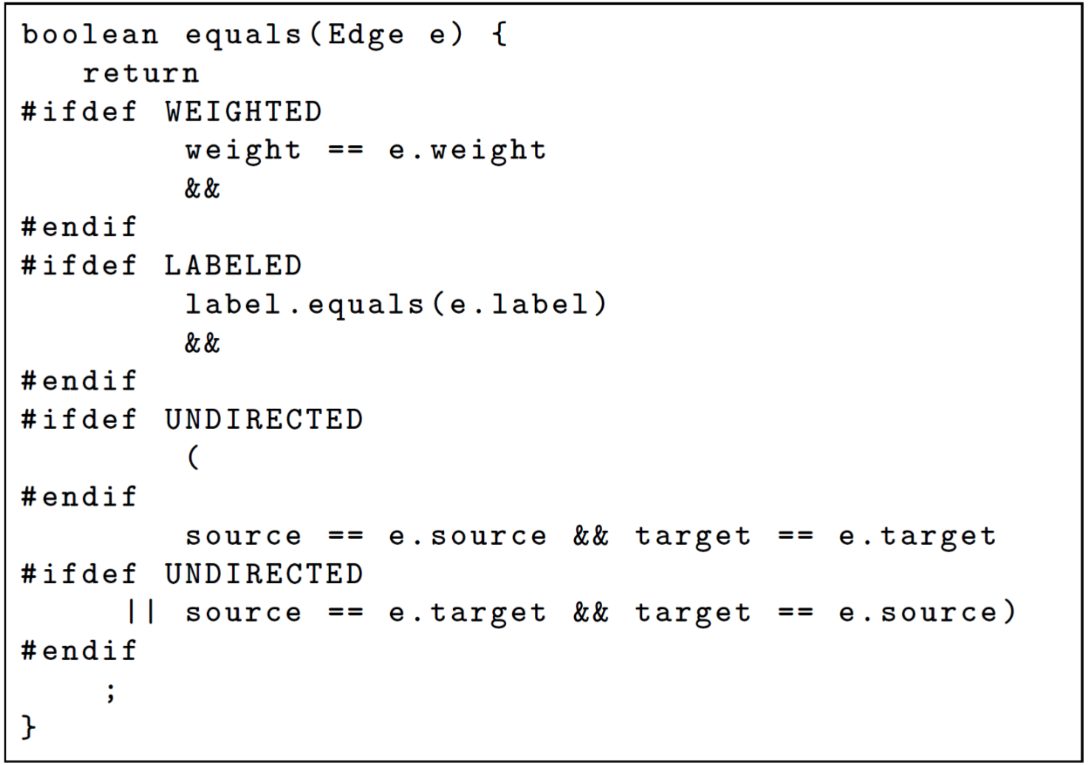
■ Graph
- ☑ Undirected
- ☑ Weighted
- □ Labeled

■ Graph
- □ Undirected
- ☑ Weighted
- ☑ Labeled

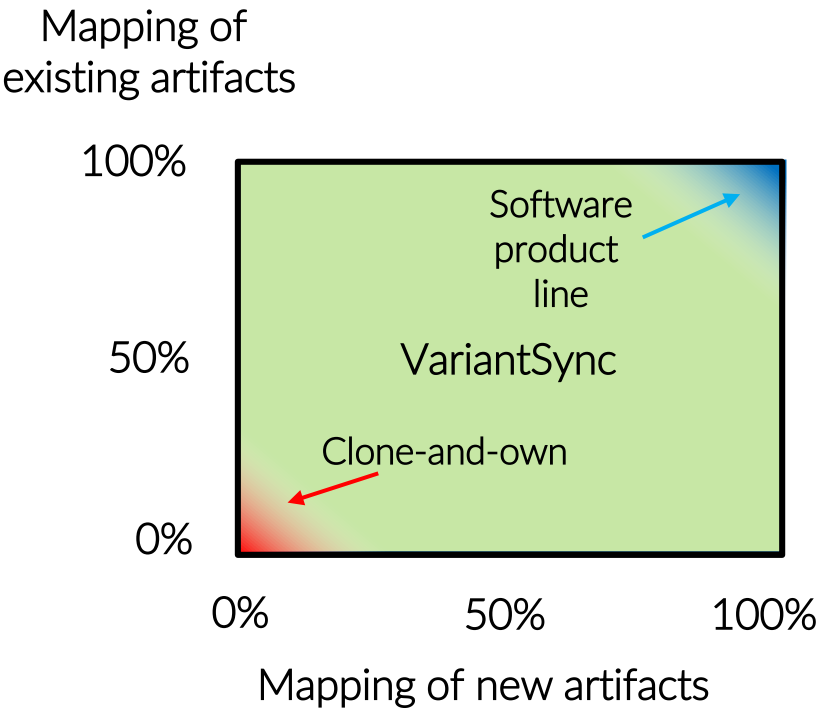


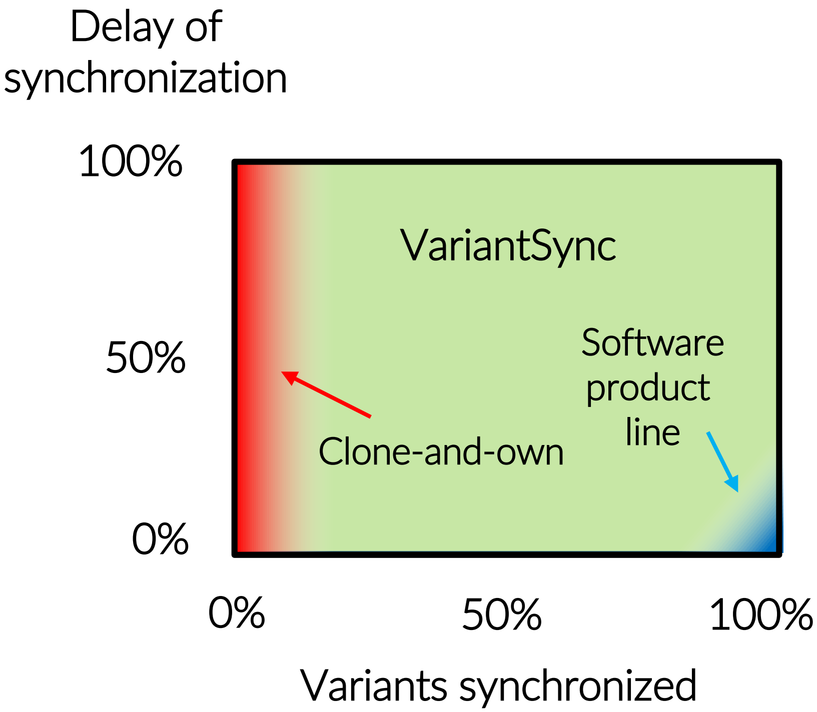
Features

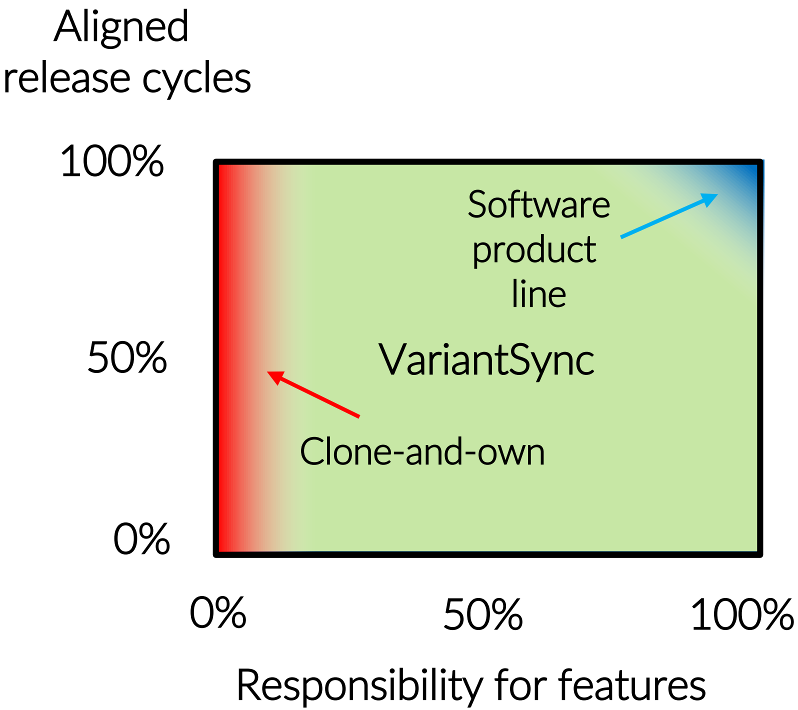
But Why Not a Software Product Line?



```
boolean equals(Edge e) {
   return
#ifdef WEIGHTED
        weight == e.weight
        &&
#endif
#ifdef LABELED
        label.equals(e.label)
        &&
#endif
#ifdef UNDIRECTED
        (
#endif
        source == e.source && target == e.target
#ifdef UNDIRECTED
     || source == e.target && target == e.source)
#endif
    ;
}
```


Mapping of existing artifacts
100%
50%
0%
Software product line
VariantSync
Clone-and-own
0%
50%
100%
Mapping of new artifacts


Delay of synchronization
100%
50%
0%
VariantSync
Software product line
Clone-and-own
0%
50%
100%
Variants synchronized


Aligned
release cycles
100%
50%
0%
Software
product
line
VariantSync
Clone-and-own
0%
50%
100%
Responsibility for features

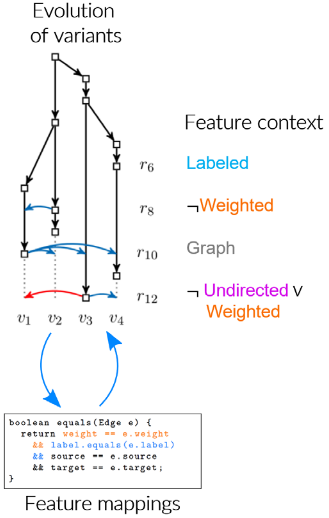
Bridging the Gap Between Clone-and-Own and Software Product Lines

Timo
Kehrer

Thomas
Thüm

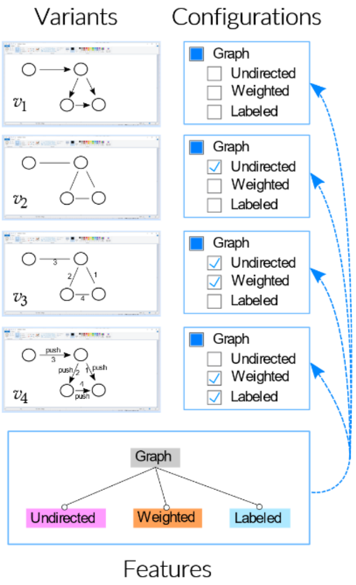
Alexander
Schultheiß

Paul
Maximilian
Bittner



```
boolean equals(Edge e) {
  return weight == e.weight
    && label.equals(e.label)
    && source == e.source
    && target == e.target;
}
```

Feature mappings

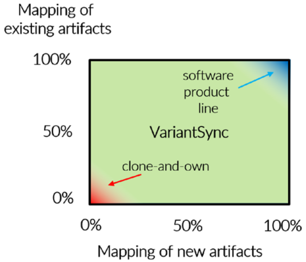


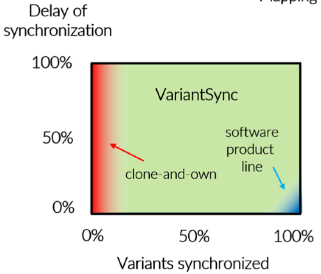
v_1: ■ Graph, □ Undirected, □ Weighted, □ Labeled

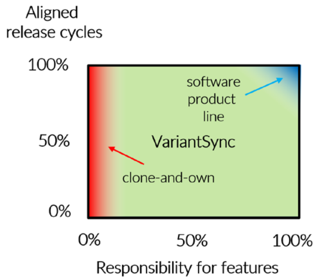
v_2: ■ Graph, ☑ Undirected, □ Weighted, □ Labeled

v_3: ■ Graph, ☑ Undirected, ☑ Weighted, □ Labeled

v_4: ■ Graph, □ Undirected, ☑ Weighted, ☑ Labeled


Mapping of
existing artifacts
Mapping of new artifacts
software
product
line
VariantSync
clone-and-own


Delay of
synchronization
Variants synchronized
VariantSync
software
product
line
clone-and-own


Aligned
release cycles
Responsibility for features
software
product
line
VariantSync
clone-and-own