An Empirical Study of Long-Lived Code Clones

Dongxiang Cai
Hong Kong University of Science and Technology
Miryung Kim*
The University of Texas at Austin

Fundamental Approaches in Software Engineering 2011

Synopsis

We hypothesize that the benefit of clone removal may depend on how long clones survive in the system.

To ***selectively identify clones to refactor,*** we investigate the characteristics of ***long-lived clones***.

Finding

We study 33.25 years of clone evolution history from 7 large projects.

The ***evolutionary characteristics of clones are better indicators*** for a clone survival time than spatial characteristics.

Outline

- **Motivation**
- A Study of Long-Lived Clones
  - Clone Evolution History Extraction.
  - Feature Vector Extraction and Correlation Analysis
  - Survival Time Prediction Model
- Limitations
- Related Work and Conclusion

Motivation

- Code cloning is ***not necessarily harmful*** **[Cordy et al. Kapser & Godfrey, Kim et al. LaToza et al.]**
- ***Refactoring may not be always*** applicable to or ***beneficial*** for code clones. **[FSE'05 Kim et al.]**

Motivation

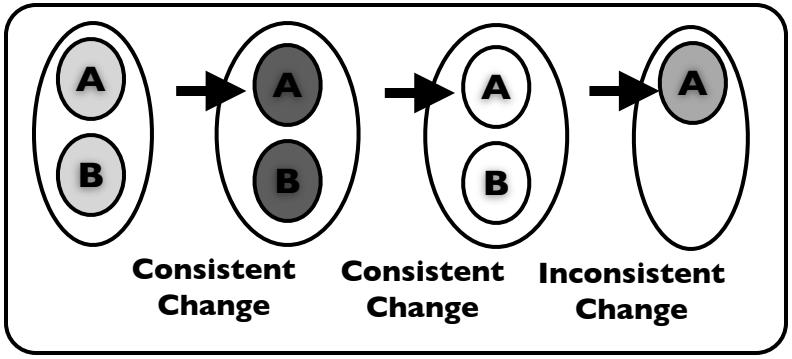
- In our study of clone genealogies **[FSE'05 Kim et al.]**, we found that
  - some clones never change during evolution.
  - some clones disappear in a short amount of time due to divergent changes.
  - some clones stay in a system for a long time and undergo similar updates repetitively.

It is crucial to selectively identify clones to refactor.

Outline

- Motivation
- **A Study of Long-Lived Clones**
  - Clone Evolution History Extraction.
  - Feature Vector Extraction and Correlation Analysis
  - Survival Time Prediction Model
- Limitations
- Related Work and Conclusion

Study Overview



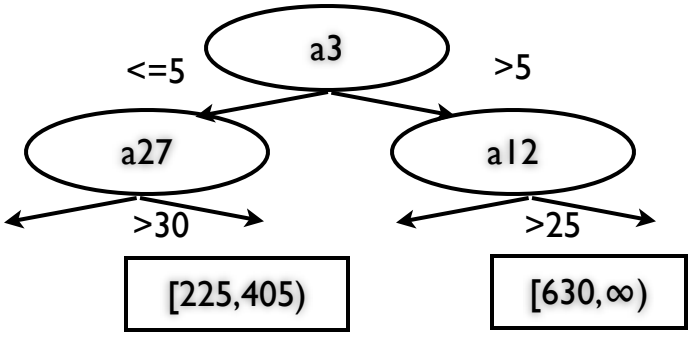
Step 1. Clone Genealogy Construction

	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

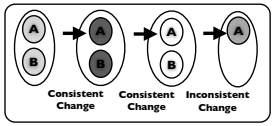
Step 2. Feature Vector Extraction

	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

Step 3. Correlation Analysis

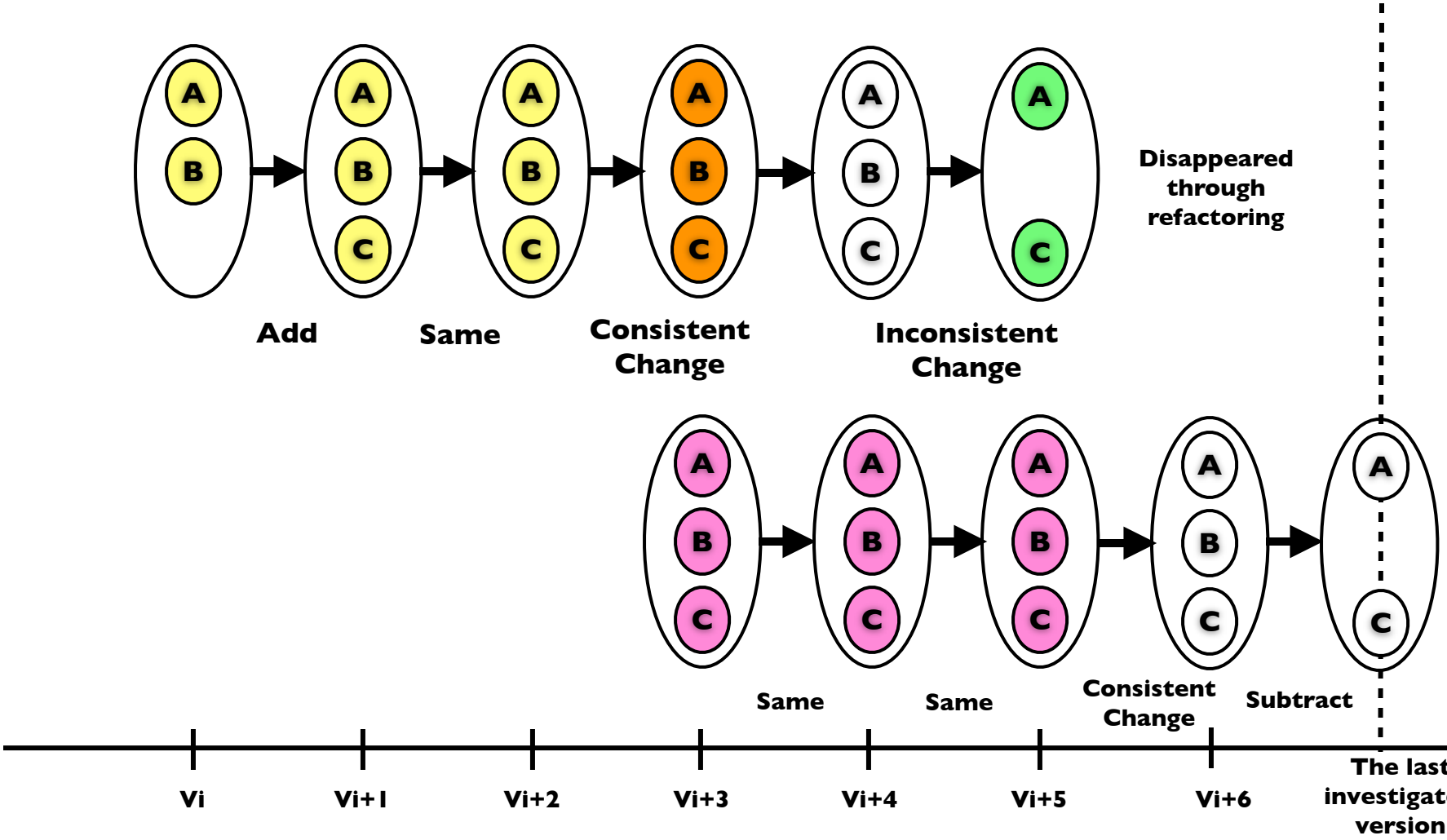


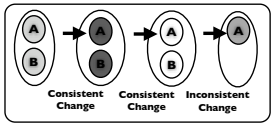
Step 4. Clone Survival Time Prediction Model



Clone Genealogy

[FSE '05 Kim et al.]

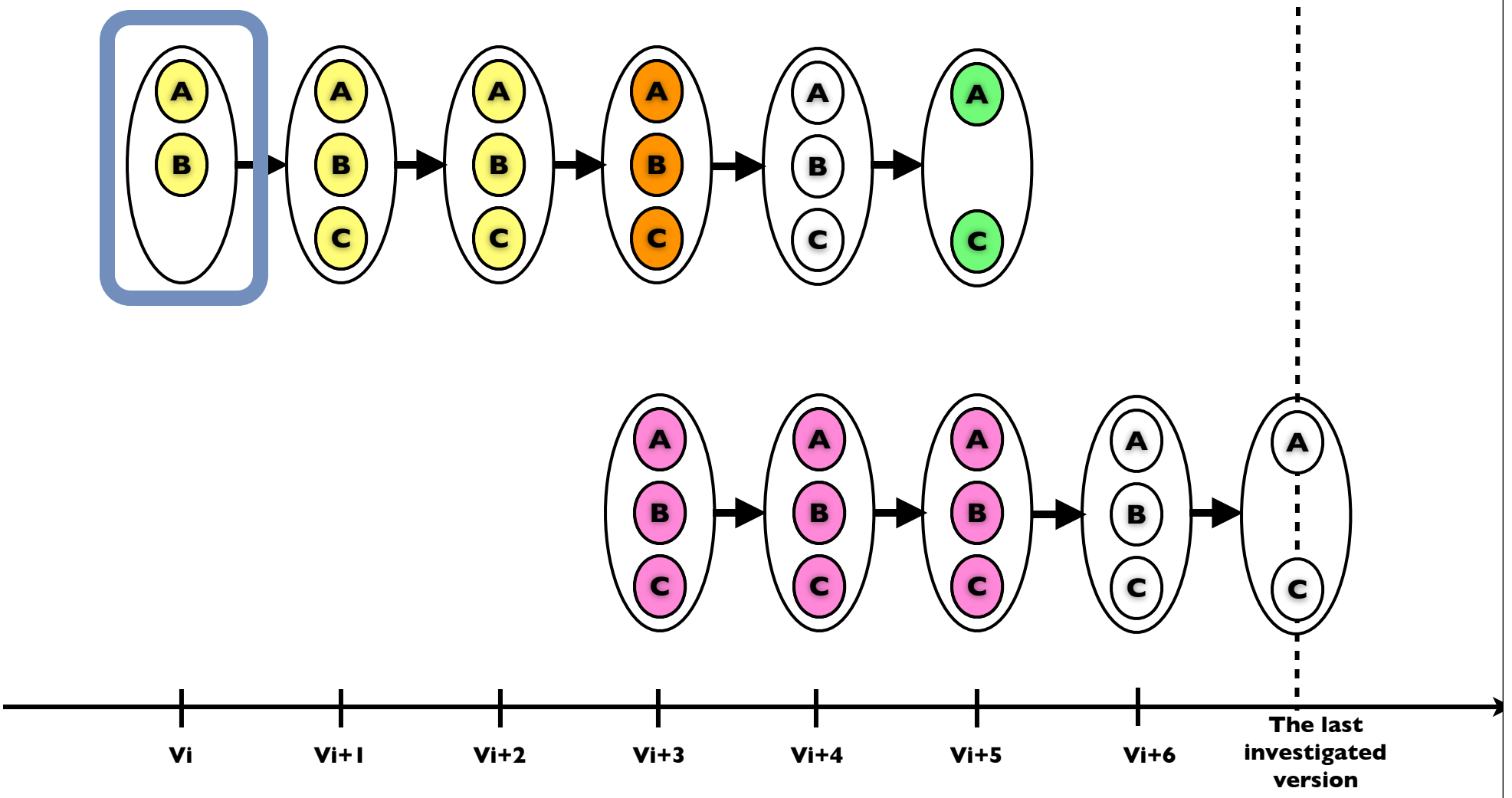


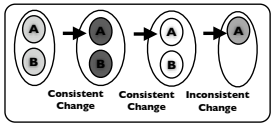


Clone Genealogy

clone group

[FSE '05 Kim et al.]

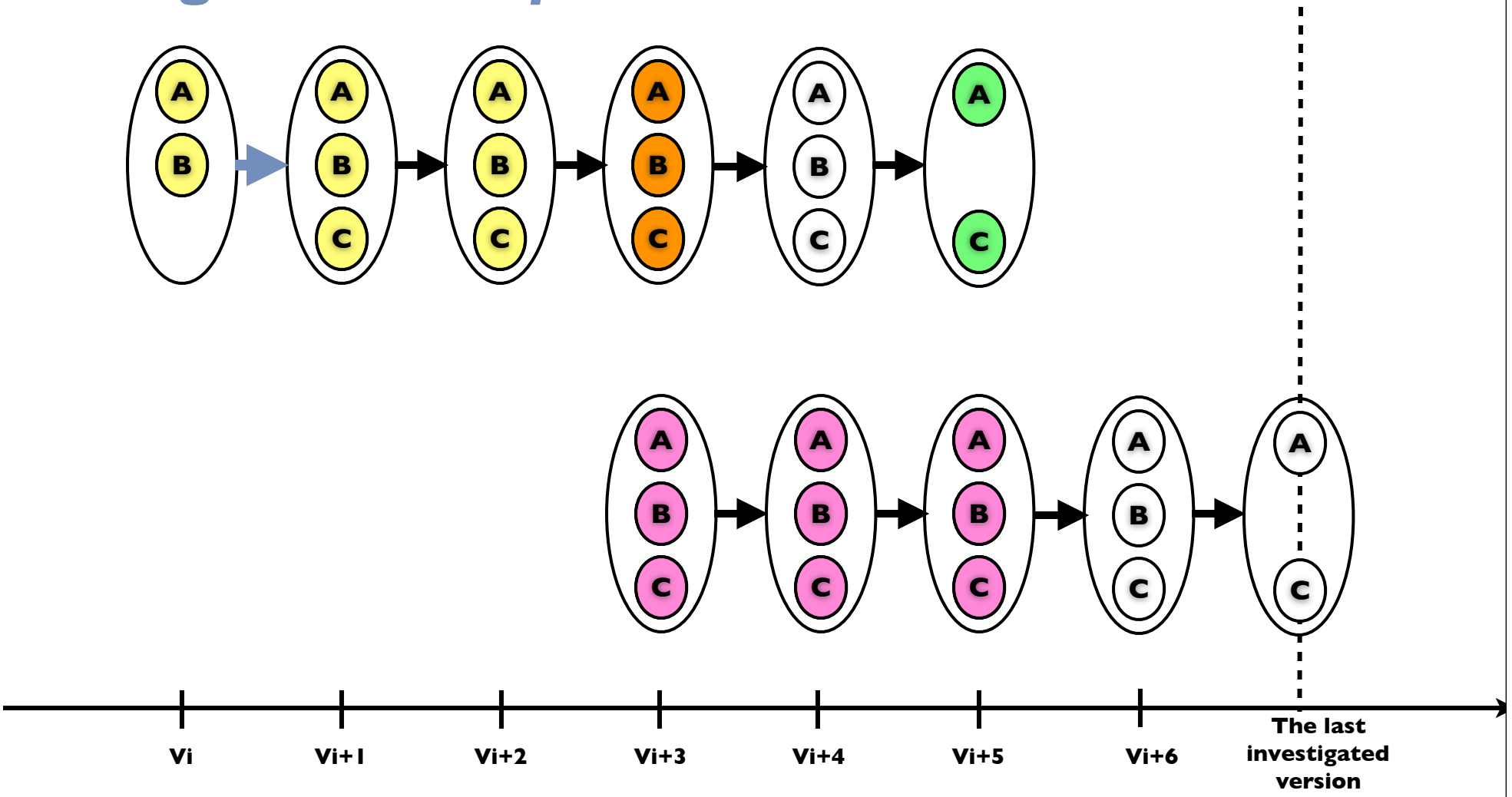


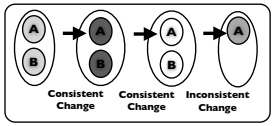


Clone Genealogy

cloning relationship

[FSE '05 Kim et al.]

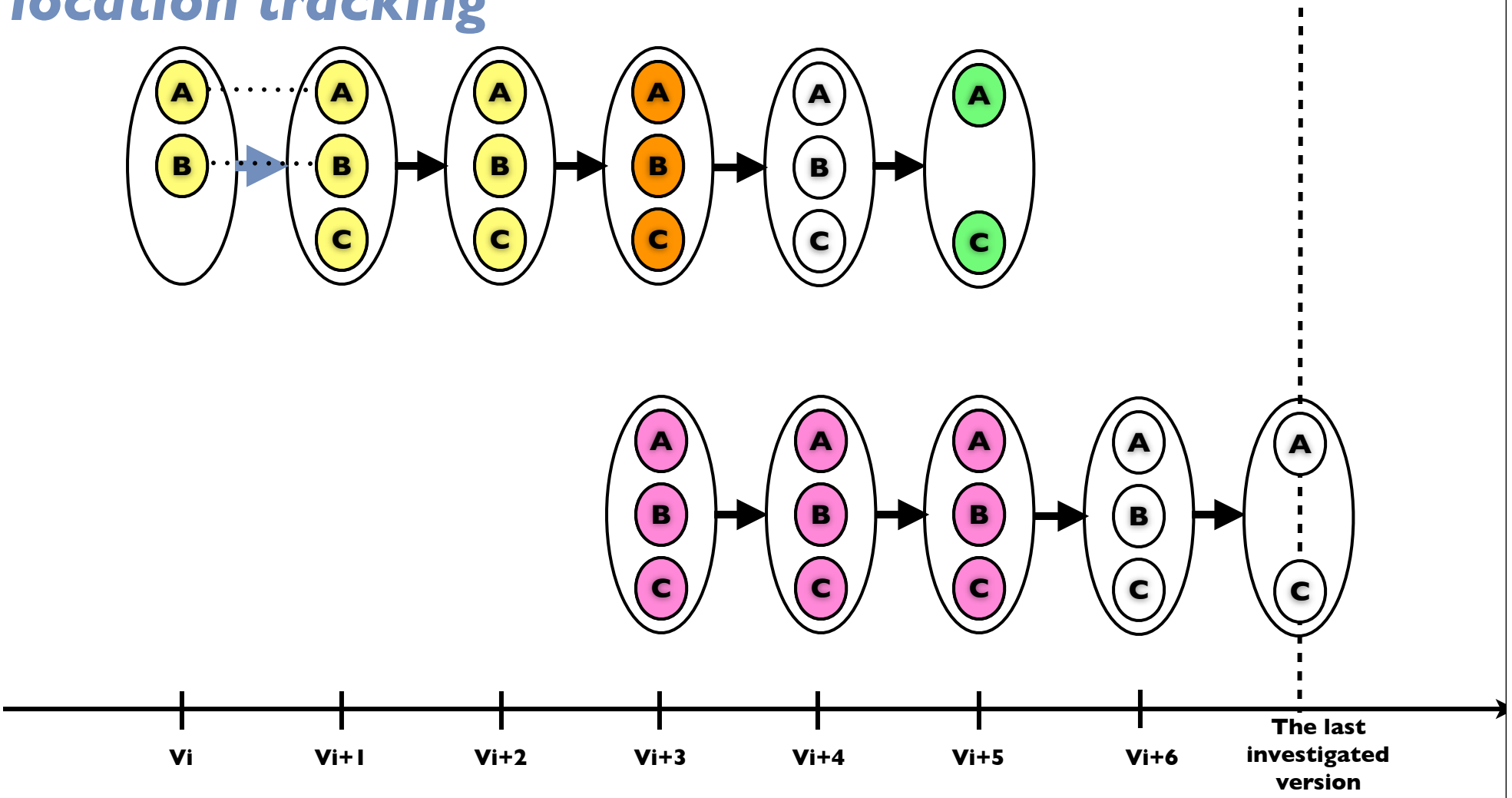


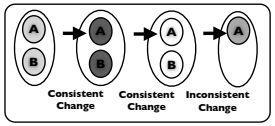


Clone Genealogy

[FSE '05 Kim et al.]

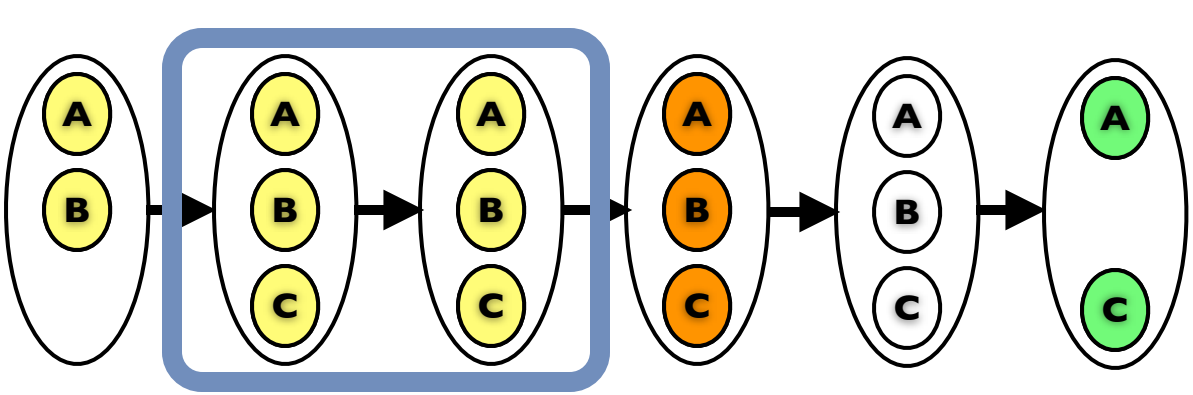
location tracking



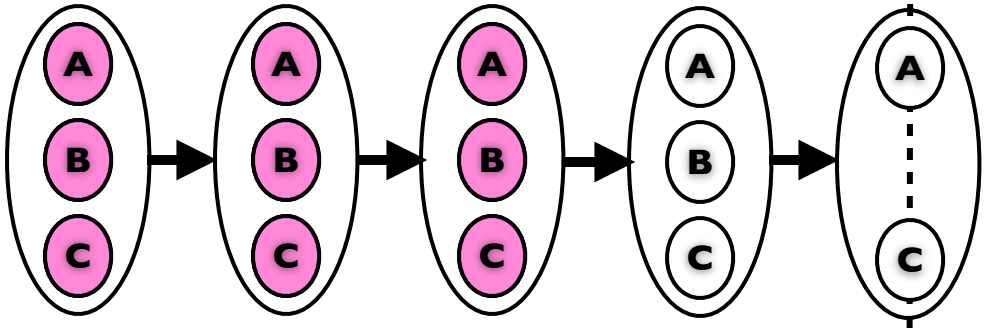


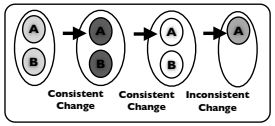
Clone Genealogy

[FSE '05 Kim et al.]



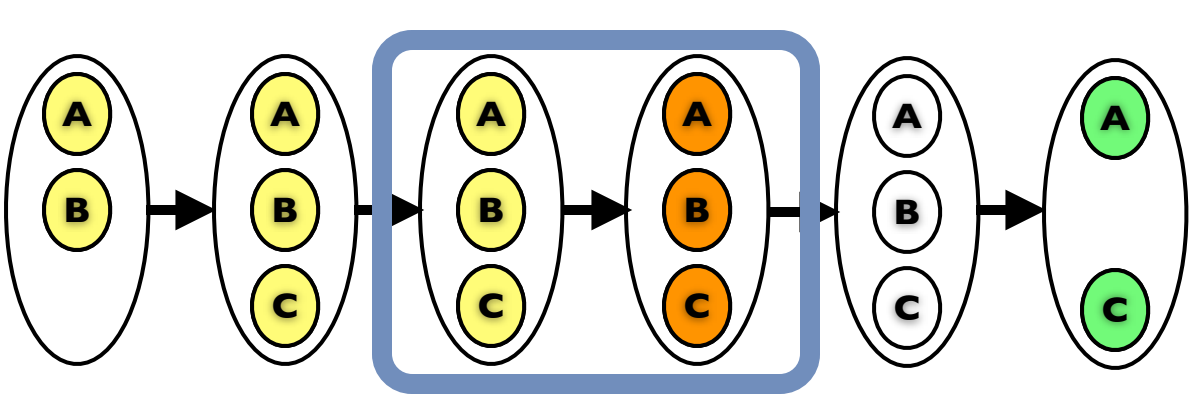
same



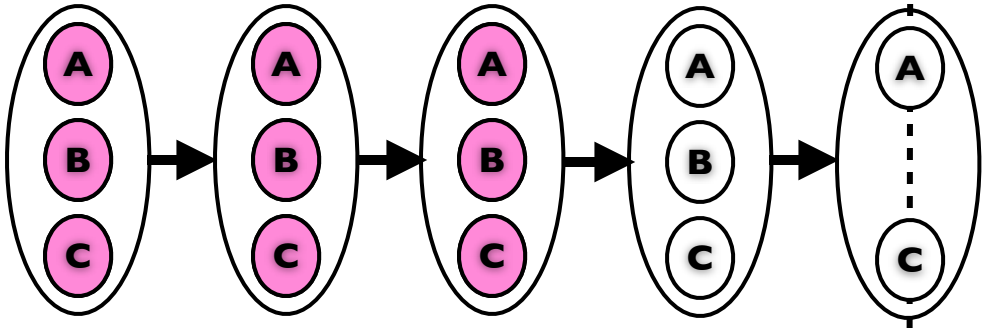


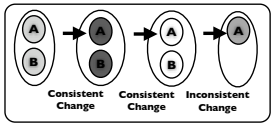
Clone Genealogy

[FSE '05 Kim et al.]



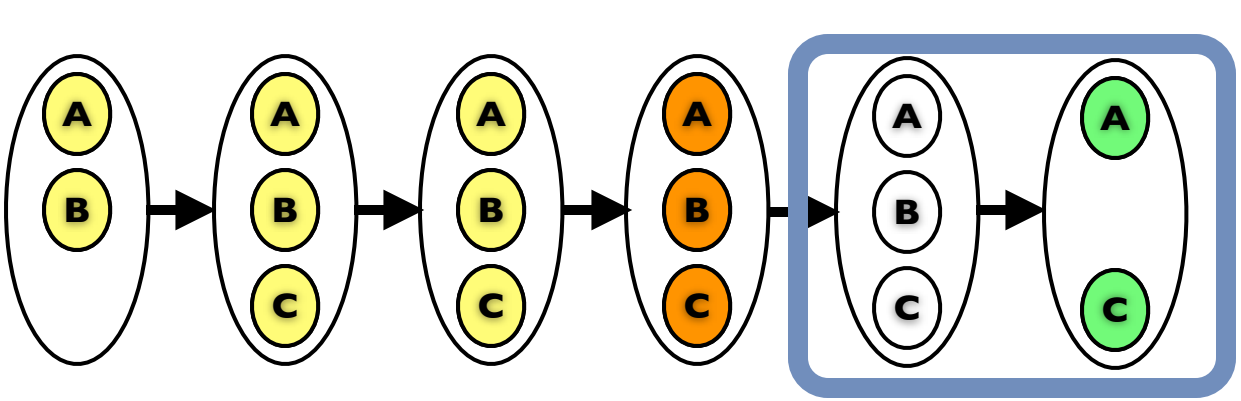
consistent change



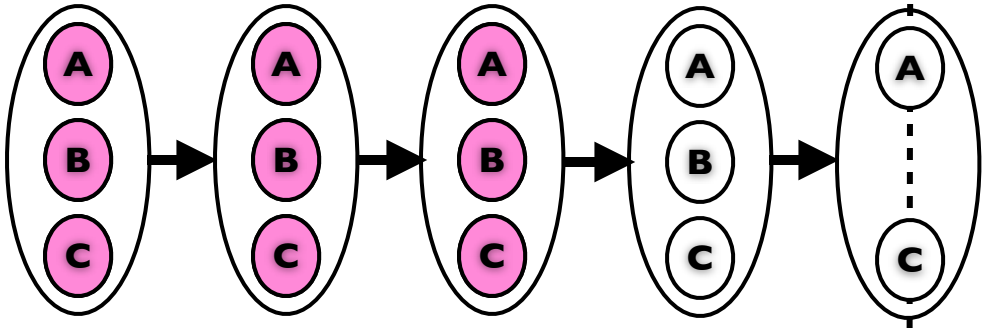


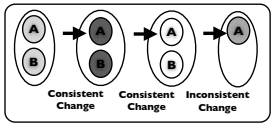
Clone Genealogy

[FSE '05 Kim et al.]



inconsistent change

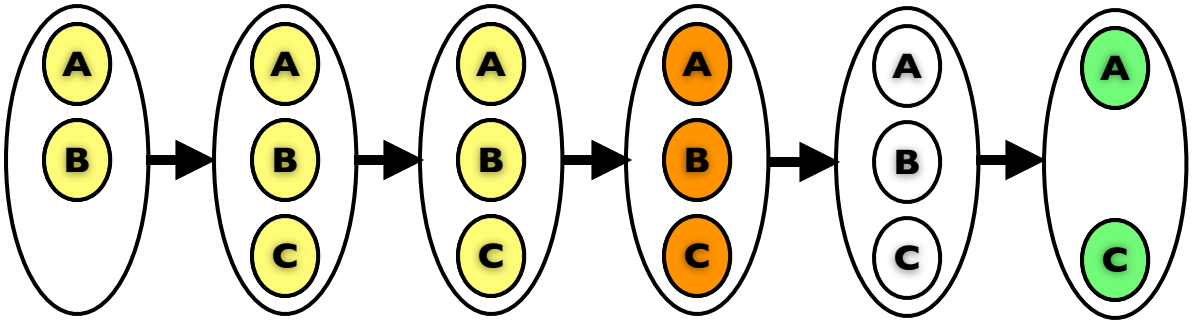




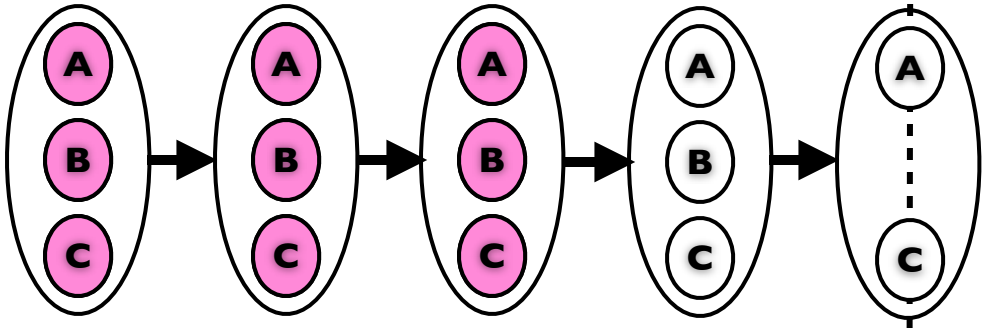
Dead vs. Alive

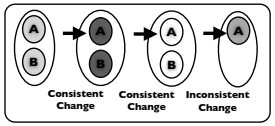
[FSE '05 Kim et al.]

Dead Genealogy: Disappeared at the age of 5 versions



Alive Genealogy: Present in the last version with the age of 4 versions

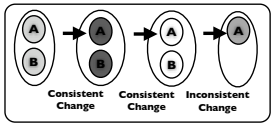




Clone Genealogy Construction [FSE '05 Kim et al.]

Given multiple versions of a program V_k for $1 \leq k \leq n$

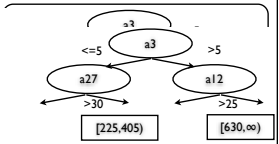
- find clone groups in each version using CCFinder (threshold setting: 40 tokens)
- find cloning relationships among clone groups of V_i and V_{i+1} using CCFinder (threshold setting: 0.8 similarity)
- map clones of V_i and V_{i+1} using diff based algorithm.
- separate each connected component of cloning relationships (a clone genealogy)
- identify clone evolution patterns in each genealogy



Data Sets

project	LOC	duration (months)	# of check-ins	# of versions
Columba	80448~194031	42 months	420	420
Eclipse	216813~424210	92 months	13790	21
hadoop	226643~315586	14 months	410	18
hadoop pig	46949~302316	33 months	906	8
HTMLunit	35248~279982	94 months	5850	22
jEdit	84318~174767	91 months	3537	26
JFreeChart	284269~316954	33 months	916	7

In total, we studied 7 large projects, 33.25 years of release history.



Clone Genealogies

(min token=40, sim th=0.8)

project	Total	Alive	Dead	Dead with age>0
Columba	556	452	104	102
Eclipse	3190	1257	1933	1826
hadoop	3094	627	2467	455
hadoop pig	3302	2474	828	422
HTMLunit	1029	500	529	425
jEdit	654	232	422	245
JFreeChart	1733	1495	238	219

Outline

- Motivation
- **A Study of Long-Lived Clones**
  - Clone Evolution History Construction
  - **Feature Vector Extraction and Correlation Analysis**
  - Survival Time Prediction Model
- Limitations
- Related Work and Conclusion

	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

Feature Vector Extraction

- We extracted 35 attributes to encode the characteristics of a clone genealogy.

1. evolutionary characteristics (9 attributes)
2. spatial characteristics (3 attributes)
3. physical dispersion (21 attributes)
4. developers (2 attributes)

- class label: clone survival time (in days)

	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

1. Evolutionary Characteristics

- # of modifications in the container files
- # of *consistent change* patterns
- A relative timing of *consistent change* pattern with respect to the age of a genealogy
  - Similarly, 6 attributes are defined for *add*, *subtract*, and *inconsistent update* patterns.

	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

2. Spatial Characteristics

- Total LOC of clones
- # of clones in each group
- The average size of a clone in terms of LOC
- We use information from the last version.

	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

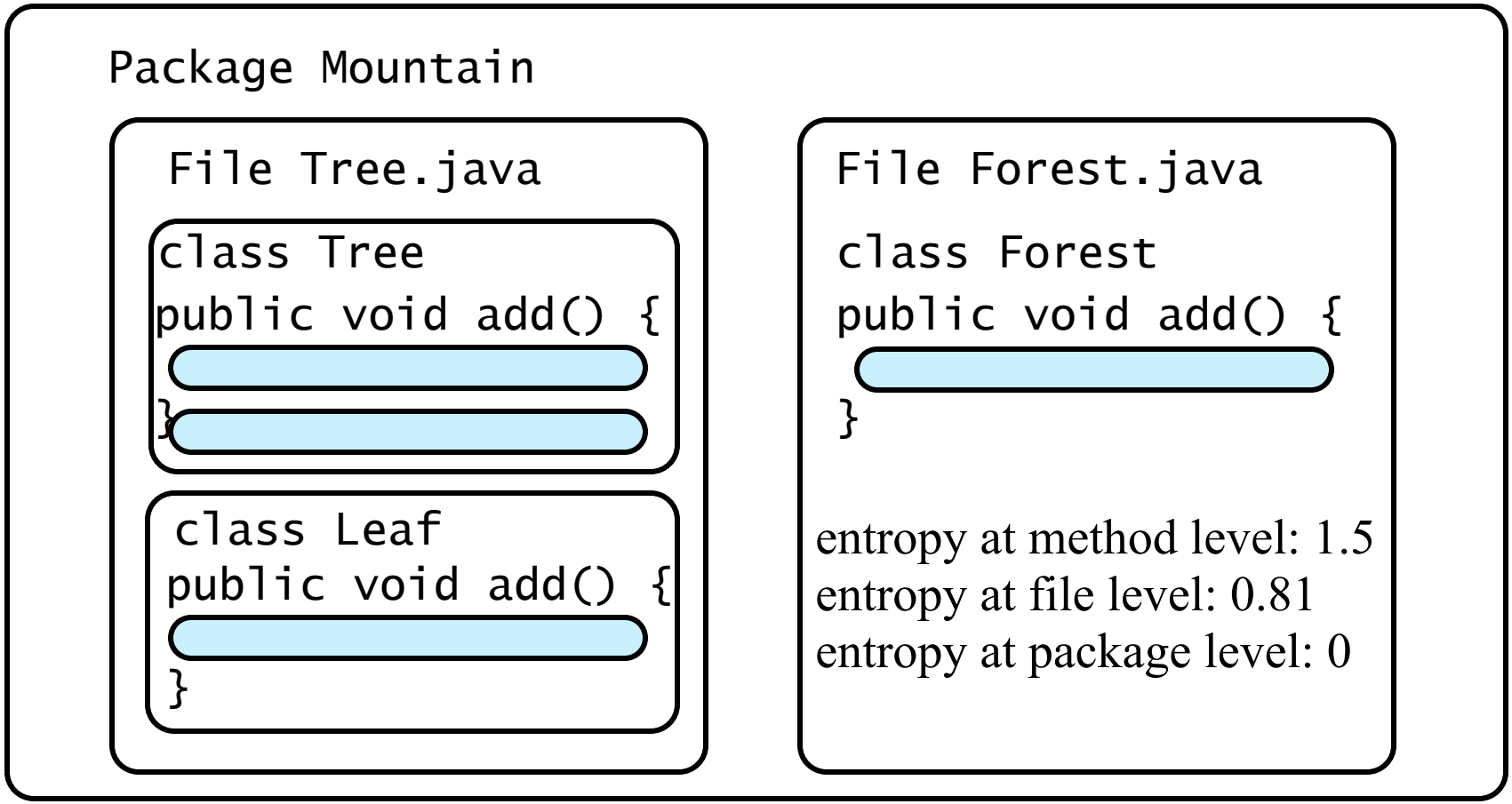
3. Physical Dispersion

- The farther clones are located from one another, the harder it is to find and refactor them.
- We encoded physical distribution of clones at different levels (method, class, file, package, and directory) in terms of **entropy:**

$$entropy = \sum_{i=1}^{n} -p_i log(p_i)$$

	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

Entropy Example



	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

4. Developer Characteristics

- # of developer involved in maintaining clones.
- The distribution of file modifications in terms of developer.
  - The higher the entropy is, more developers equally contributed to clone maintenance.

	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

Pearson's Correlation Analysis

- We measure Pearson's correlation coefficient between each attribute and a clone genealogy survival time *(class label)*.
- We ranked attributes in terms of correlation strength.

	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

Weak Correlation

- The size of clones is not strongly correlated with a clone survival time (ρ=0.009).
- The number of clones in each group is not strongly correlated with a clone survival time (ρ=0.016).
- The physical dispersion of clones is not strongly correlated with a clone survival time (ρ=0.023, 0.018).

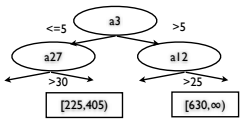
	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

Strong Correlation

- The more uniformly developers contribute to maintaining clones, the longer time it takes for clones to be removed (ρ=0.553).
- The more developers maintained clones, the longer the survival time (ρ=0.528).
- The longer it has been since the last addition or deletion of a clone, the longer it takes for the clone to be removed (ρ=0.481, 0.479).

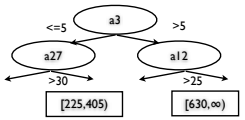
Outline

- Motivation
- **A Study of Long-Lived Clones**
  - Clone Evolution History Extraction
  - Feature Vector Extraction and Correlation Analysis
  - **Survival Time Prediction Model**
- Limitations
- Related Work and Conclusion



Predicting Clone Survival Time

- We create a training data set by categorizing each clone genealogy's clone survival time into five categories: *very short-lived*, *short-lived*, *normal*, *long-lived*, and *very long-lived*
- This process requires finding an unbiased binning scheme.



Binning Scheme

- uniform binning scheme

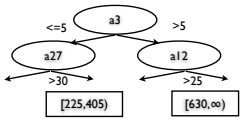
χ=50

[0, 50)	[50, 100)	[100, 150)	[150, 200)	[200, 250)

- incremental binning scheme
  - $bin_i = bin_{i-1} + 0.5 * \chi(i+1)$

χ=50

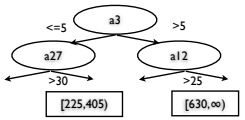
[0, 50)	[50, 125)	[125, 225)	[225, 350)	[350, ∞)



Selected Binning

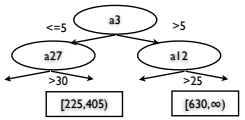
project	# of vectors	survival time (days)	# of genealogies for each category
Columba	102	1.1~1222.2	[0,60):18, [60,120):8, [120,180):9, [180,240):16,(240+):51
Eclipse	1826	687.1~2010	[0,90):204, [90,225):423, [225,405):340, [405,630):510, [630+):349
hadoop common	455	34~585	[0,40):324, [40,100):66, [100,180):16, [180,280):33, [280+):16
hadoop pig	422	30~536.9	[0,40):131, [40,100):92, [100,180):97, [180,280):31, [280+):72
HTMLunit	425	6.9~2122.4	[0,60):125, [60,150):119, [150,270):63, [270,420):24, [420+): 94
jEdit	245	13.3~2281.7	[0,70):22, [70,175):321, [175,315):31, [315,490):22, [490+): 139
JFreeChart	219	11.1~415	[0,50):37, [50,125):2, [125,225):104, [225,350):38, [350+):38

(varied χ from 30 to 100 in an increment of 10 and selected the binning with the highest entropy)



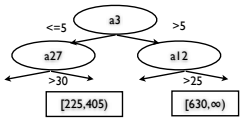
Building Prediction Model

- We then used various classifiers in the Weka tool kit to build survival time prediction models.
- We use 10 fold cross validation to measure precision and recall.



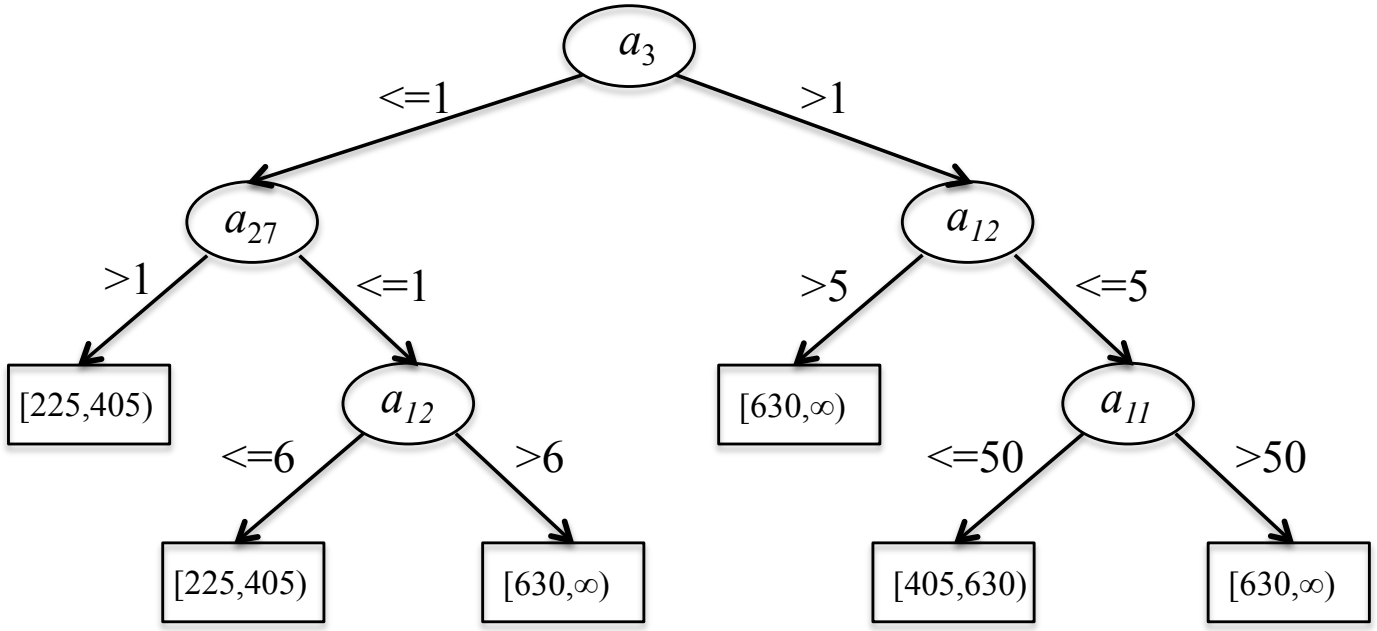
J48 Prediction Model

project	Precision	Recall
Columba	58.1%	58.8%
Eclipse	79.4%	79.3%
hadoop common	74.5%	78.0%
hadoop pig	79.1%	79.1%
HTMLunit	73.3%	73.6%
jEdit	62.0%	65.7%
JFreeChart	68.2%	70.3%
Total	75.7%	76.5%



Prediction Model in JDT

- a_3 : The number of *add* evolution patterns.
- a_{11}: The number of times that files containing clones were modified.
- a_{12}: The number of developers involved in maintaining clones.
- a_{27}: The number of unique methods that clones in the last version are located.



Study Limitations

- CCFinder does find non-contiguous clones.
- We used release snapshots as opposed to check-in snapshots.
- We did not consider the dispersion of clones in a class inheritance hierarchy or how easy to refactor those clones.
- We have not investigated the impact of threshold settings in clone genealogy construction.

Related Work

- Identification of refactoring opportunities **[Higo et al. Koni-N'Sapu, Balazinska et al. Tsantalis and Chatzigeorgiou et al, etc.]**
- Studies about code cloning practice **[Cordy et al. Kapser & Godfrey, Kim et al. LaToza et al.]**
- Clone evolution analysis **[Lague et al. Krinke, Aversano et al. Balint et al.]**
- Classification of code clones **[Kapser and Godfrey, Bellon et al.]**

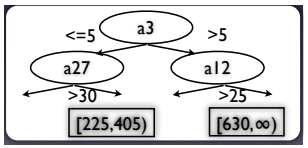
Conclusion

- As a first step to ***selectively identify clones to refactor,*** we studied 33 years of clone evolution history in 7 large projects.
- We found that the ***size, number, and physical dispersion of clones are weakly correlated*** with a clone survival time.
- On the other hand, the number of ***developers*** who worked on clones and the frequency and recency of changes to clones have ***stronger correlation*** with their survival time.

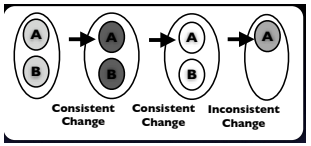
Acknowledgment

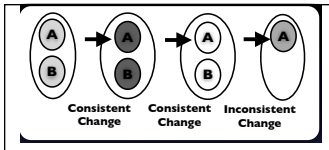
This research is in part supported by
National Science Foundation, CCF-1043810.

- back up



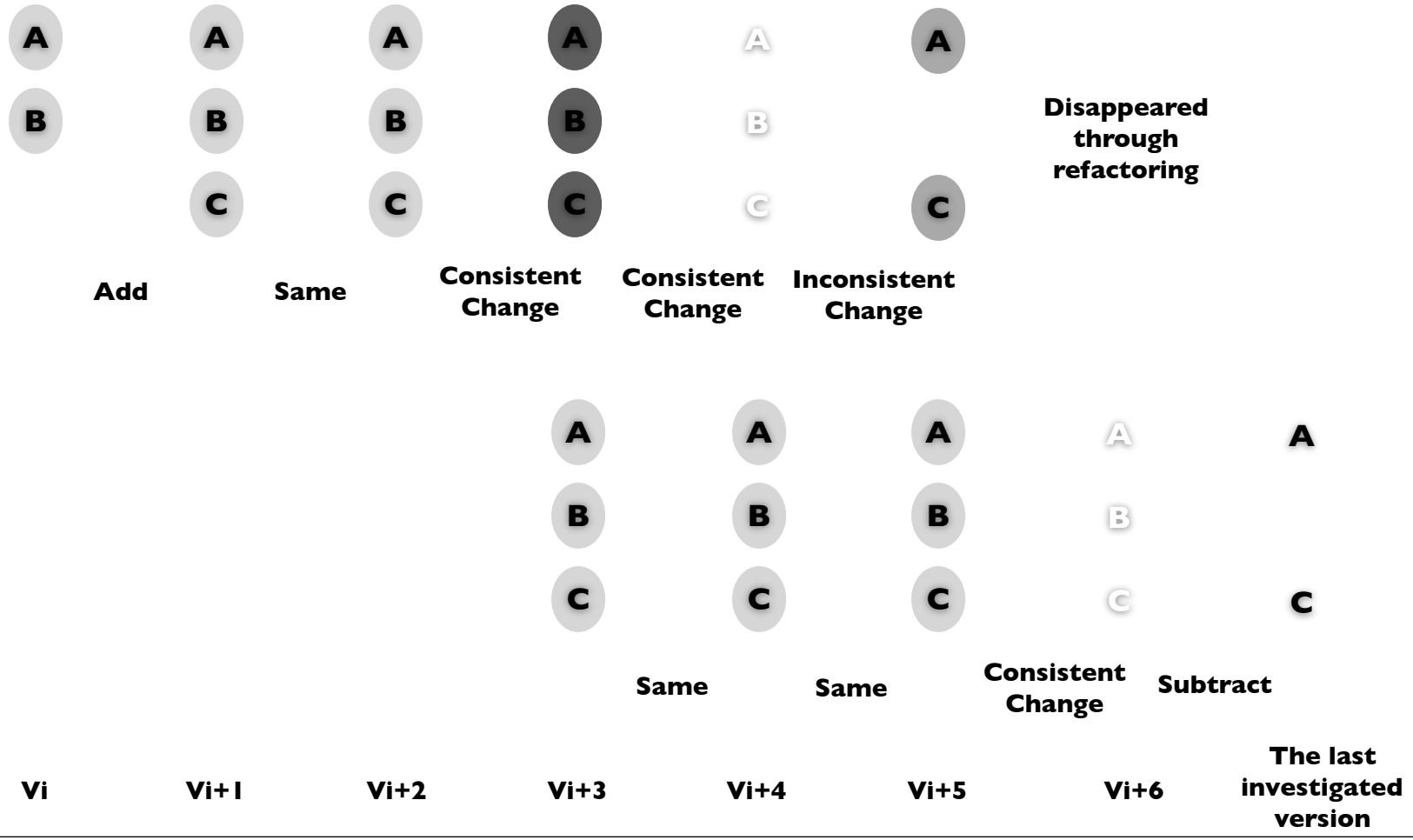
	a1	a2	a3	survival time
G1	1	4	1	12 days
G2	3	5	2	101 days

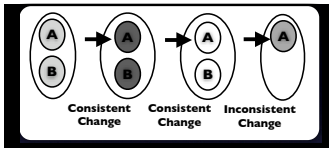




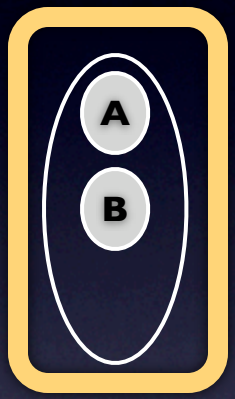
Clone Genealogy

[FSE '05 Kim et al.]

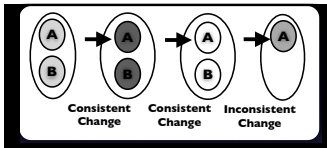




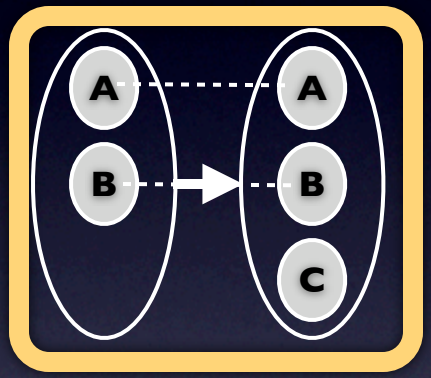
Clone Group



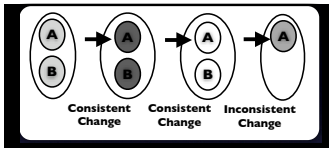
*A **clone group** is a set of clones considered equivalent according to a clone detector.*



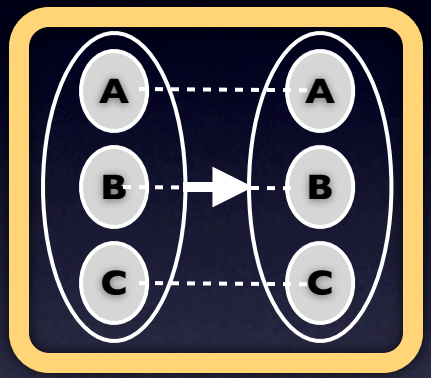
Clone Evolution Patterns



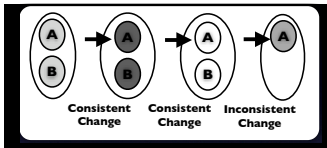
Add *means that at least one code snippet is newly added to the clone group.*



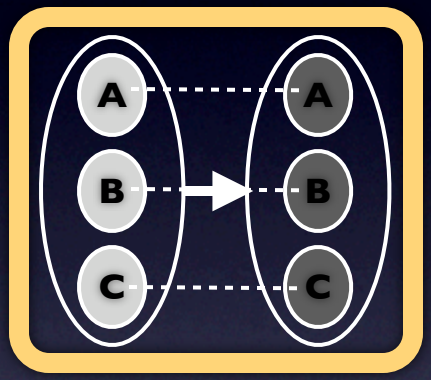
Clone Evolution Patterns



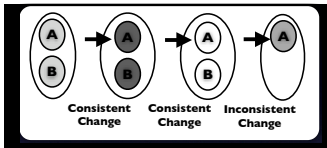
Same *means all code snippets in the new version's clone group did not change from the old version's clone group.*



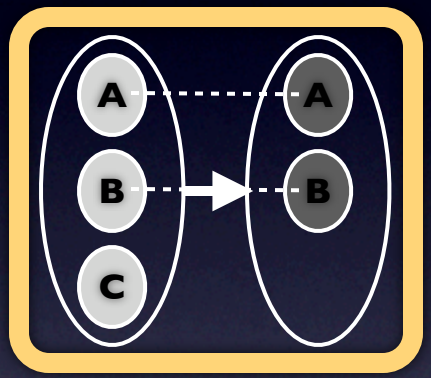
Clone Evolution Patterns



Consistent change *means all code snippets in the old version's clone group have changed consistently; thus they all belong to the new group.*



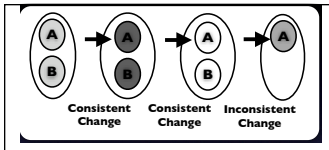
Clone Genealogy [FSE '05 Kim et al.]



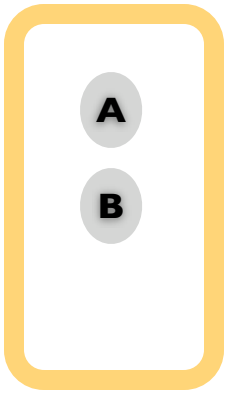
Inconsistent change *means at least one code snippet in the old version's clone group have changed inconsistently; thus it no longer belongs to the same group.*

Dispersion Attributes

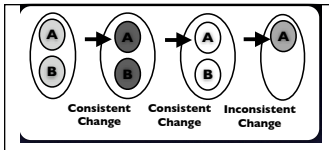
- # of methods in which clones in the *last* version are located
- *entropy* at the method level for clones in the *last* version
- # of methods in which clones in *all* versions are located
- *entropy* at the method level for clones in *all* versions
  - Similarly, 16 attributes are defined for class, file, package, directory levels.



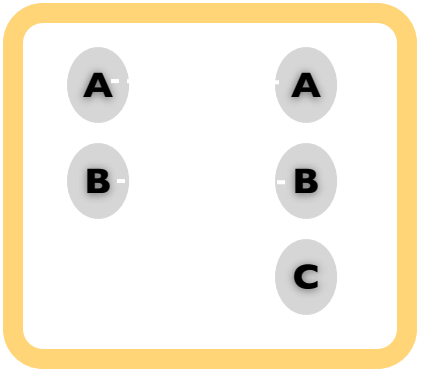
Clone Group



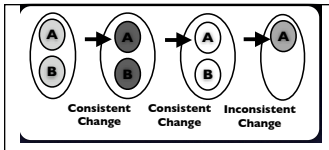
*A **clone group** is a set of clones considered equivalent according to a clone detector.*



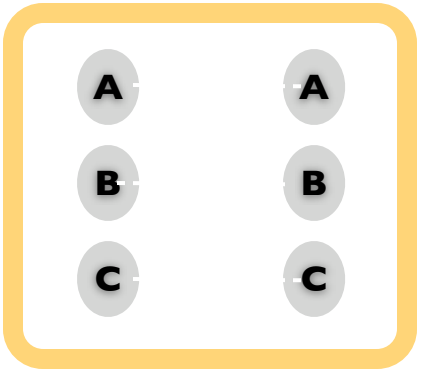
Clone Evolution Patterns



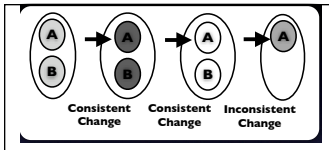
Add *means that at least one code snippet is newly added to the clone group.*



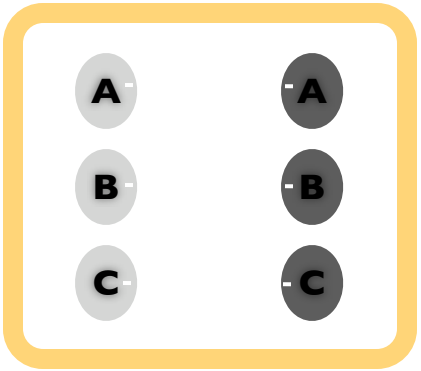
Clone Evolution Patterns



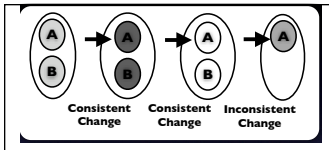
Same *means all code snippets in the new version's clone group did not change from the old version's clone group.*



Clone Evolution Patterns

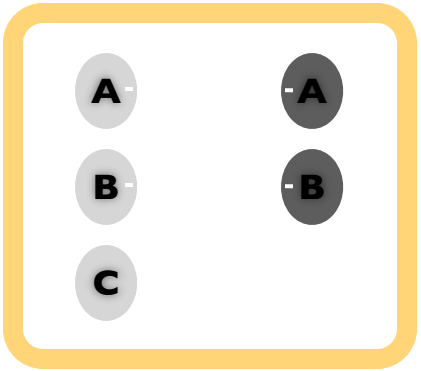


Consistent change *means all code snippets in the old version's clone group have changed consistently; thus they all belong to the new group.*



Clone Genealogy

[FSE '05 Kim et al.]



Inconsistent change *means at least one code snippet in the old version's clone group have changed inconsistently; thus it no longer belongs to the same group.*