Can *Requirements Dependency Network* Be Used as Early Indicator of *Software Integration Bugs*?

Junjie Wang, Juan Li, Qing Wang, Da Yang, He Zhang, Mingshu Li
Institute of Software Chinese Academy of Sciences, Beijing, China

Agenda

- Background & Motivation
- Methodology
- Results
- Conclusions
- Future work

Agenda

- **Background & Motivation**
- **Methodology**
- **Results**
- **Conclusions**
- **Future work**

Background — dependency

Requirements dependency

Requirements relate to and affect each other (Similar_to, Constraints, Precondition)

Software quality

1. Software complexity
2. Cohesion and coupling

1. Change propagation
2. Create technical dependency

中国科学院软件研究所 基础软件国家工程研究中心 National Engineering Research Center of Fundamental Software Institute of Software, Chinese Academy of Sciences

Background — network analysis

- Network analysis
  - Originate from social sciences
  - Network measures
    - Signify the node's position with respect to other nodes
    - Centrality
  - Software engineering
    - Call and data flow relationship, Collaboration when fixing bugs, Communication through e-mail, etc.

Motivation

Requirements dependency

Network analysis

Integration bugs

(Connections or interactions between software components)

Can *Requirements Dependency Network* Be Used as Early Indicator of *Software Integration Bugs*?

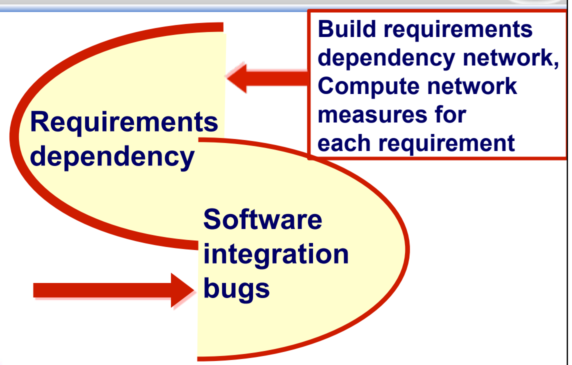
Motivation — research questions

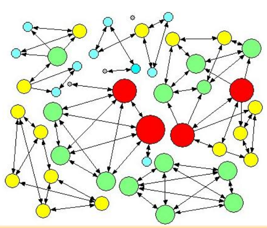
- Correlation
  - Q1: Do network measures on requirements dependency network *correlate with* the number of bugs?
- Prediction
  - Q2: Can network measures on requirements dependency network be used to *predict* the number of bugs?

Agenda

- Background & Motivation
- Methodology
- Results
- Conclusions
- Future work

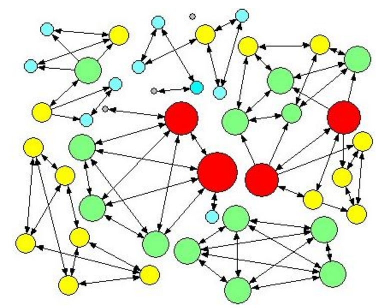
Methodology





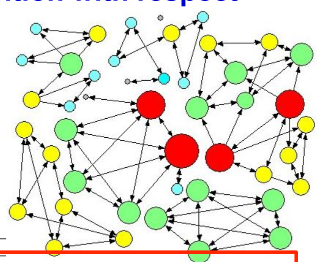
Methodology — network construction

- **Network : model requirements dependency**
  - Nodes : requirements
  - Edges : dependencies between two requirements
  - Precondition, Constraint and Similar to
    - Change propagation



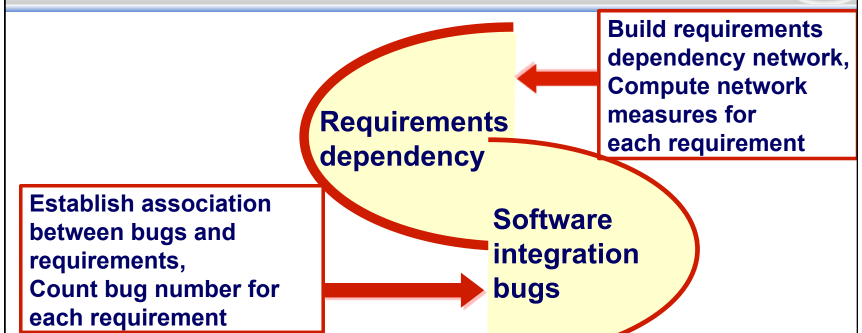
Methodology — network measures

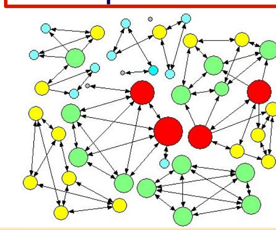
- **Network Measures**
  - **Signify the requirements' position with respect to other requirements**



Measure	Description
Centrality	
Degree	The number of nodes that are directly connected to a node.
Closeness	The total length of all shortest paths from a node to all other nodes.
Reachability	The weighted number of nodes that can be reached from a node. The weigh is 1, 1/2, 1/3 for nodes that are 1, 2 or 3 steps away.
Betweenness	The number of shortest paths between other nodes that a node occurs.
Structural holes	
EffiSize	The number of nodes that are connected to a node minus the average number of ties between these nodes.
Efficiency	The EffiSize normalized by size of the network.
Constraint	The extent to which the node is limited in option to reach other nodes.
Hierarchy	Concentration of constraint in the global network.

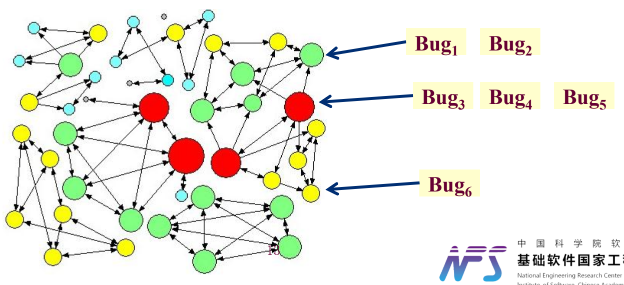
Methodology



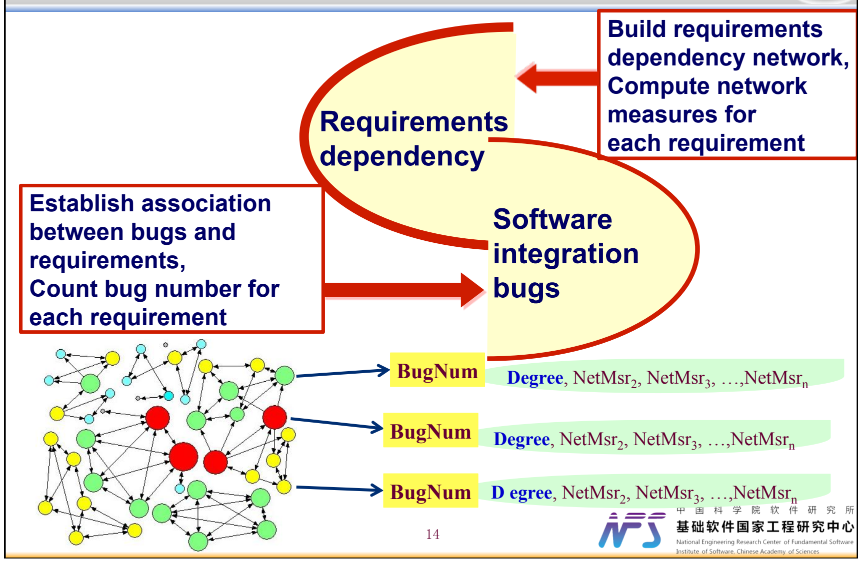


Methodology — bug association

- Establish association between bugs and requirements
  - Three practitioners with different background
  - Three steps, interviews to discuss differences
- Why not automated support?

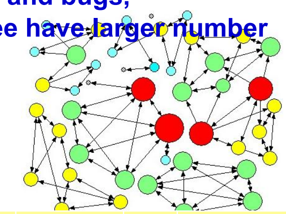


Methodology



Methodology — correlation

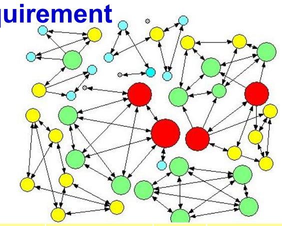
- Q1: Do network measures *correlate with the number of bugs?*
  - **Spearman correlation analysis**
  - **Relationship between requirements and bugs, e.g. requirements with higher degree have larger number of bugs**



$BugNum_1$	Req_1	⟷	Req_1	$Degree_1$	Net Msr_{12}	Net Msr_{13}	…	Net Msr_{1n}
$BugNum_2$	Req_2	⟷	Req_2	$Degree_2$	Net Msr_{22}	Net Msr_{23}	…	Net Msr_{2n}
	……							
$BugNum_x$	Req_x	⟷	Req_x	$Degree_x$	Net Msr_{x2}	Net Msr_{x3}	…	Net Msr_{xn}

Methodology — prediction

- Q2: Can network measures be used to *predict the number of bugs?*
  - **Regression analysis**
  - **Predict bug number for each requirement**



?

$BugNum_1$	Req_1	⟷	Req_1	$Degree_1$	Net Msr_{12}	Net Msr_{13}	…	Net Msr_{1n}
$BugNum_2$	Req_2	⟷	Req_2	$Degree_2$	Net Msr_{22}	Net Msr_{23}	…	Net Msr_{2n}
	……							
$BugNum_x$	Req_x	⟷	Req_x	$Degree_x$	Net Msr_{x2}	Net Msr_{x3}	…	Net Msr_{xn}

Methodology — prediction (Cont'd)

- Q2: Can network measures be used to *predict the number of bugs?*
  - ***Data splitting*: 2/3 to build model and 1/3 to measure its efficacy, 50 random splits**
  - ***Multi-collinearity:* correlation between degree and closeness is 0.48, Variable Inflation Factor (VIF)**
  - ***Evaluate explanative power:* R-Square**
  - ***Evaluate predictive power:* Mean Squared Error (MSE) and Spearman correlation coefficient**

Agenda

- **Background & Motivation**
- **Methodology**
- **Results**
- **Conclusions**
- **Future work**

Results — experimental projects

- **Experimental projects**
  - **Medium-sized software organization in China**
  - **Achieved CMMI maturity level 4**
  - **Qone product, released in 2004 and 13 versions**
  - **Two versions for experiment**

	Project A	Project B
Number of requirements	308	334
(Compared with last version)		
Number of added requirements	11	38
Number of modified requirements	26	28
Number of deleted requirements	13	2
Lines of code	550K	580K
Number of integration bugs	732	418

Results — correlation

- Q1: Do network measures *correlate with the number of bugs?*
  - **Spearman correlation analysis**
  - **Relationship between requirements and bugs,**
    e.g. requirements with higher degree have larger number of bugs
- Q2: Can network measures be used to *predict the number of bugs?*
  - **Regression analysis**
  - **Predict bug number for each requirement**

Results — high correlation coefficient

- [-1, +1], negative – positive
- +/- 0.5, significant correlation
  +/- 0.7, strong correlation
- Significant correlations for most measures
- Central requirements are more bug-prone

TABLE IV: Spearman correlation between network measures of HybridNetwork and bug number for Project A. All correlations are significant at the 0.01 level (p <0.01).

	In	Out	Un
Ego Network			
Size	0.811	0.794	0.831
Ties	0.502	0.502	0.527
Pairs	0.619	0.658	0.673
Density	0.668	0.612	0.650
AvgDist	0.440	0.468	0.478
WeakComp	0.776	0.745	0.772
nWeakComp	0.625	0.529	0.511
TwoStepReach	0.789	0.769	0.785
ReachEfficency	0.717	0.675	0.646
Brokerage	0.475	0.504	0.518
nBrokerage	0.611	0.579	0.602
EgoBetweenness	0.465	0.504	0.509
nEgoBetweenness	0.337	0.584	0.607
Global Network			
Centrality measures			
Degree	0.811	0.794	0.831
Closeness	0.734	0.757	0.621
Reachability	0.763	0.770	0.769
Betweenness			0.508
Structural holes measures			
EffSize			0.783
Efficiency			0.514
Constraint			0.438
Hierarchy			-0.455

Results — correlation for different dependencies

- Among Precondition, Constraints and Similar_to, *Constraints* dependency exerts highest correlation
- Reason:
  - Involve call and data flow dependency relationship when propagating to code level

TABLE VI: Spearman correlation between network measures of non-HybridNetwork and bug number. All correlations are significant at the 0.01 level (p <0.01).

	PreNetwork	ConNetwork	SimNetwork
	Project A		
Size-In	0.241	***0.616***	0.276
Ties-In	0.201	***0.537***	0.242
Density-Un	0.189	***0.437***	0.189
TwoStepReach-Un	0.266	***0.570***	0.275
WeakComp-Out	0.302	***0.600***	0.269
Brokerage-Out	0.198	***0.448***	0.251
EgoBetweenness-In	0.182	***0.458***	0.229
Closeness-In	0.241	***0.448***	0.393
Reachability-Out	0.310	***0.623***	0.281
Betweenness-Un	0.185	***0.462***	0.231
EffSize-Un	0.236	***0.518***	0.231
Constraint-Un	0.163	***0.427***	0.189
	Project B		
Size-In	0.145	***0.558***	0.360
Ties-In	0.152	***0.544***	0.351
Density-Un	0.188	***0.329***	0.287
TwoStepReach-Un	0.204	***0.613***	0.358
WeakComp-Out	0.213	***0.547***	0.342
Brokerage-Out	0.163	***0.462***	0.297
EgoBetweenness-In	0.131	***0.304***	0.198
Closeness-In	0.147	***0.397***	0.290
Reachability-Out	0.227	***0.551***	0.358
Betweenness-Un	0.133	***0.302***	0.201
EffSize-Un	0.179	***0.456***	0.324
Constraint-Un	0.124	***0.241***	0.200

Results — prediction

- Q1: Do network measures *correlate with the number of bugs?*
  - Spearman correlation analysis
  - Relationship between requirements and bugs,
    e.g. requirements with higher degree have larger number of bugs
- Q2: Can network measures be used to *predict the number of bugs?*
  - Regression analysis
  - Predict bug number for each requirement

中 国 科 学 院 软 件 研 究 所
基础软件国家工程研究中心
National Engineering Research Center of Fundamental Software
Institute of Software, Chinese Academy of Sciences

Results — predictive performance

- ***Low*** **MSE(Mean Squared Error)**
  - High accuracy
  - difference between predicted and actual
- ***High*** **Spearman correlation coefficient**
  - High sensitivity
  - Increase in predicted accompanied by increase in actual

(a) R-Square (Project A) (b) R-Square (Project B)

(c) MSE (Project A) (d) MSE (Project B)

(e) Spearman (Project A) (f) Spearman (Project B)

Results — significant predictors

- Significant predictors for bug number
  - WeakComp-Out, Closeness-In and Betweenness-Un
    - Over 80% of the 50 random splits, p-value for these measures are less than 0.05
  - Central requirements are more bug-prone

Agenda

- Background & Motivation
- Methodology
- Results
- Conclusions
- Future work

Conclusions — correlation

- **Answer for Q1:** Do network measures *correlate with the number of bugs?*
  - **Most of network measures significantly correlate with bug number**

Centrality measures			
Degree	0.811	0.794	0.831
Closeness	0.734	0.757	0.621
Reachability	0.763	0.770	0.769

  - **Constraint type of dependency contributes more to bug indication**

Conclusions — prediction

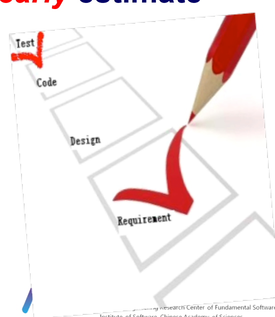
- **Answer for Q2:** Can network measures be used to *predict the number of bugs?*
  - **Network measures can predict bug number with high accuracy and sensitivity**

		Min	Max	Mean
Project A	R-Square	0.681	0.891	0.757
	MSE	0.62%	1.86%	1.08%
	Spearman	0.570	0.835	0.704
Project B	R-Square	0.526	0.863	0.710
	MSE	0.62%	1.59%	0.98%
	Spearman	0.607	0.880	0.777

  - **WeakComp-Out, Closeness-In and Betweenness-Un are the significant predictors**

Conclusions — contribution

- Requirements dependency network can be used as an early indicator of software integration bugs
  - Resource allocation and decision making
- Utilizing information already present in requirements phase and provide *early* estimate regarding software quality



Agenda

- Background & Motivation
- Methodology
- Results
- Conclusions
- Future work

中国科学院软件研究所
基础软件国家工程研究中心
National Engineering Research Center of Fundamental Software
Institute of Software, Chinese Academy of Sciences

Future work

- Replicate the study with other commercial and open source software projects
- Conduct cross-project prediction

Future work (Cont'd)

- Explore other requirement-related indicators for software bugs
  - Implementation relationship between requirements and program elements
  - Dependency weighting schema
  - Communication structure within requirement teams
- What is it that makes the central requirements more bug-prone?

中国科学院软件研究所 基础软件国家工程研究中心
National Engineering Research Center of Fundamental Software
Institute of Software, Chinese Academy of Sciences

Contact:

wangjunjie@itechs.iscas.ac.cn