

Revealing Software Development Work Patterns with PR-Issue Graph Topologies - Supplementary Materials

Cleudson R. B. de Souza
Universidade Federal do Pará
Belém, Brazil
cleudson.desouza@acm.org

Emilie Ma
The University of British Columbia
Vancouver, Canada
contact@emilie.ma

Jesse Wong
The University of British Columbia
Vancouver, Canada
nami5504@gmail.com

Dongwook Yoon
The University of British Columbia
Vancouver, Canada
yoon@cs.ubc.ca

Ivan Beschastnikh
The University of British Columbia
Vancouver, Canada
bestchai@cs.ubc.ca

ACM Reference Format:

Cleudson R. B. de Souza, Emilie Ma, Jesse Wong, Dongwook Yoon, and Ivan Beschastnikh. 2024. Revealing Software Development Work Patterns with PR-Issue Graph Topologies - Supplementary Materials. In *Original paper published at the Companion Proceedings of the 32nd ACM Symposium on the Foundations of Software Engineering (FSE '24), July 15–19, 2024, Porto de Galinhas, Brazil*. ACM, New York, NY, USA, 6 pages. <https://doi.org/10.1145/nnnnnnn.nnnnnnn>

1 PROJECT SCRIPTS

The scripts, queries, and raw data used to generate the statistics in the paper are available, with documentation, [in this GitHub repository](#). Please see the repository for usage instructions and additional implementation details. See [this link](#) for the final *WorkflowsExplorer* tool.

2 GITHUB PROJECTS

See Table 1 for the list of GitHub Projects studied. This table also contains information about the number of PRs and Issues in each project.

3 FURTHER PROJECT STATISTICS

Figure 1 shows the number of components within each project and the project network's density across the number of nodes in the project. In general, it is possible to observe that we studied a wide variety of projects, i.e., good coverage of projects of various sizes and component counts. As expected, the larger the number of nodes, the larger the number of components. As also expected, the larger the number of nodes, the smaller the density since more edges are necessary to keep the density constant. Furthermore, it is important to notice the density values (y-axis) that are very small, which means that the resulting PR-Issue graphs from each project are mostly disconnected, although with variation among them.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

FSE 2024, July 2024, Porto de Galinhas, Brazil

© 2024 Association for Computing Machinery.

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM...\$15.00

<https://doi.org/10.1145/nnnnnnn.nnnnnnn>

The bolded projects represent projects that were found in the extreme 5 projects in both graphs. The extreme cases in both graphs indicate interesting repository characteristics. For example, *apache/dubbo* has a high number of components but one of the lowest densities, indicating it has many isolated or dual-node components. On the other hand, *cruffenach/CRTtoast* has a low number of components and nodes, but a relatively high density, indicating that while the project is small, it is highly interconnected. We observe a strong positive correlation (0.977) in the first graph and a moderate negative correlation (-0.494) in the second. The negative exponential trend in the density of repositories as their size increases reflects the power-law distribution of our component sizes, with many isolated and two-node components across projects.

We also aimed to contrast small and large connected components to see if there were any statistically significant metadata patterns differentiating the two. We arbitrarily defined 'small' components as those with size ≤ 10 (making up 99.66% of our dataset), and 'large' components as those with size > 100 (0.02%). We also wished to examine trends across connected components sizes over our entire dataset.

One heuristic we examined was component duration, or the time delta between the first node creation in a component and the last node update event (see Figure 2).

We observed an initial increasing trend of increasing duration (strong positive correlation of 0.954) until around component size 15, after which there was a plateau of component duration (moderate positive correlation of 0.467). Due to a very low number of samples for some component sizes, there is a large standard error of the mean.

4 CYPHER QUERIES

The following are the Cypher queries used to perform our topological analysis. These can also be found in our source code here.

As explained in section 6.10 of the submission, the process of querying *workflow types* was affected by the order in which these types were matched. In particular, because some of the topological definition of workflows overlapped, even though these work practices semantically exclude each other. To address this problem, we first queried for the Dependent PRs and hubs and removed those instances from matches of the other workflow types. This is the

Table 1: GitHub projects studied

Project Name	# of Issues	# of PRs
MithrilJS/mithril.js	1259	1218
tristanhimmelman/ObjectMapper	799	328
archriss/react-native-snap-carousel	762	167
roboquice/roboquice	279	72
mlflow/mlflow	2338	3878
Project-OSRM/osrm-backend	4146	2131
tiny-dnn/tiny-dnn	576	475
volatilityfoundation/volatility	687	137
amphp/amp	217	171
App-vNext/Polly	557	392
BurntSushi/toml	197	162
chaijs/chai	877	583
Flipboard/bottomsheet	143	70
grpc/grpc-web	657	594
jhen0409/react-native-debugger	449	260
John-Lluch/SWRevealViewController	753	70
rematch/rematch	504	447
stacktracejs/stacktrace.js	178	51
jupyterhub/jupyterhub	2117	1801
metafizzy/flickity	1142	96
mgonto/restangular	1223	269
nytimes/NYPhotoViewer	156	194
Rappzt/discord.py	2599	2297
SwipeCellKit/SwipeCellKit	333	111
transitive-bullshit/create-react-library	283	86
iron/iron	259	381
jonschlinkert/remarkable	297	118
redis/redis-rb	586	501
MagicStack/uvloop	282	195
summernote/summernote	3232	1084
tsayen/dom-to-image	345	78
varvet/pundit	396	338
go-chi/chi	421	307
kubernetes-sigs/kustomize	1916	2792
TypeStrong/ts-node	1104	594
casesandberg/react-color	514	346
cruffenach/CRTToast	113	106
duo-labs/cloudmapper	531	398
sosedoff/pgweb	313	254
zaach/jison	282	119
Activiti/Activiti	2048	1962
ag-grid/ag-grid	4602	723
cglib/cglib	112	92
deployphp/deployer	1694	1153
fantasyland/fantasy-land	168	164
rauch/slackin	220	173
tensorpack/tensorpack	1347	205
ptomasroos/react-native-scrollable-tab-view	870	308
marko-js/marko	1011	792
rlidwka/sinopia	391	101
imbrn/v8n	62	171
apache/dubbo	5310	4912
pagekit/pagekit	765	198
deepinsight/insightface	1931	119
shadowsocks/shadowsocks-manager	537	136
sdc-alibaba/SUI-Mobile	1001	65

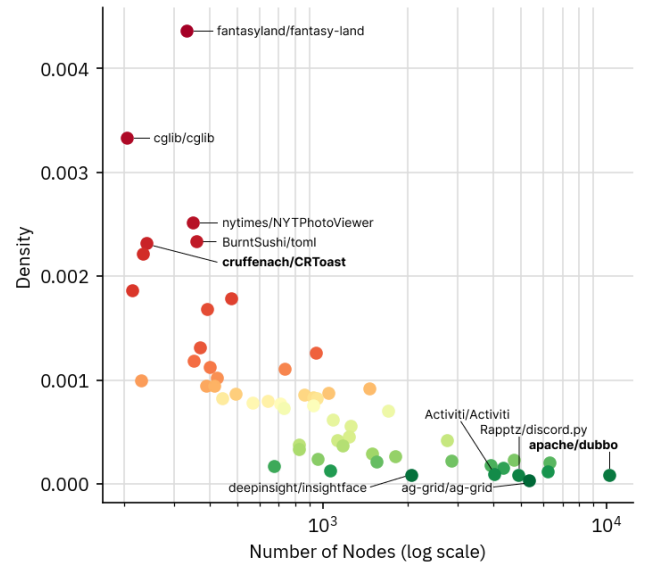
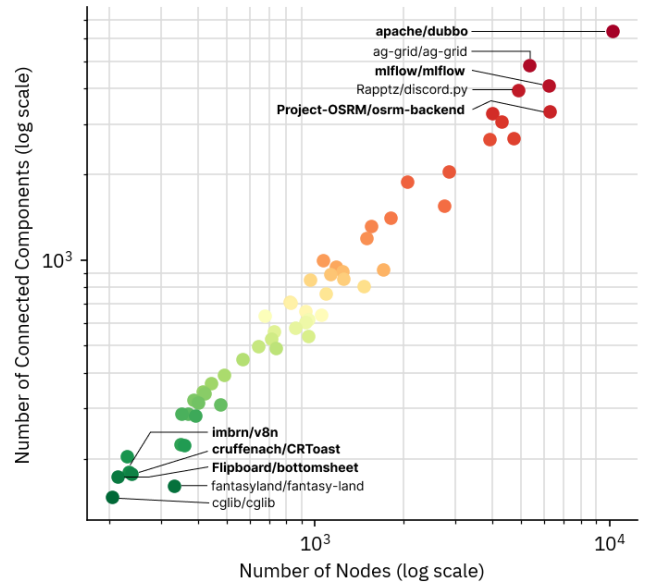


Figure 1: PR-Issue component, edge, and density information per project.

reason why a set of component IDs is presented in the beginning of the Cypher specification for in some queries.

```

call {
match (i:issue {status: "closed"})-[r {labels:"fixes"}]-(
pr:pull_request)
with i, collect(distinct pr) as pull_requests, collect (
distinct pr.user) as users, collect(r) as
match_relationships, max(pr.creation_date) as
max_date, min(pr.creation_date) as min_date

```

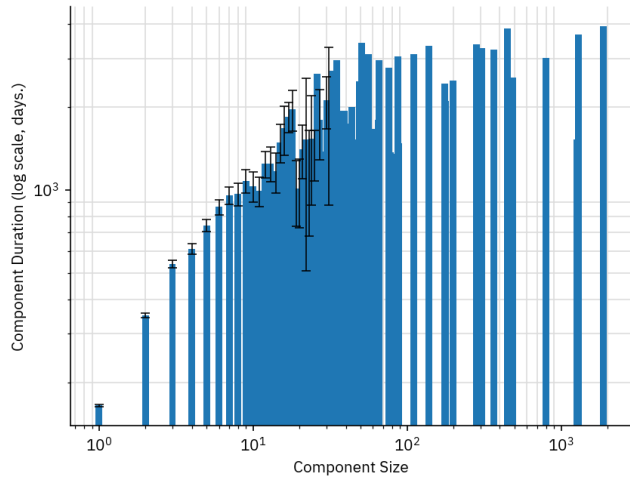


Figure 2: Component duration vs. component size across all projects.

```

257 where size([p_r in pull_requests where p_r.status="merged
258 "] = 1 and size([p_r in pull_requests where p_r.
259 status="closed"]) >= 1 and size(pull_requests) >= 2
260 and size(users) > 1 and max_date - min_date <=
261 604800 // dates are in Unix timestamps so difference
262 is in seconds
263 return collect(distinct id(i)) as known_competition
264 }
265
266 match (i:issue {status: "closed"})-[r {labels:"fixes"}]-
267 (pr:pull_request)
268 with i, collect(distinct pr) as pull_requests, collect (
269 distinct pr.user) as users, collect(r) as
270 match_relationships, max(pr.creation_date) as
271 max_date, min(pr.creation_date) as min_date,
272 known_competition
273 where size([p_r in pull_requests where p_r.status="merged
274 "] = 1 and size([p_r in pull_requests where p_r.
275 status="closed"]) >= 1 and size(pull_requests) >= 2
276 and size(users) > 1 and max_date - min_date <=
277 604800 // dates are in Unix timestamps so difference
278 is in seconds
279 call apoc.path.subgraphAll(i, {limit: case 50 > size(
280 pull_requests) when true then 50 when false then
281 size(pull_requests) + 1 end, bfs: true })
282 yield nodes, relationships
283 with i, pull_requests, match_relationships, nodes,
284 relationships, size(collect([i_node in nodes where
285 i_node.type="issue" and i_node.status="closed" and
286 i_node.number <> i.number and not id(i_node) in
287 known_competition])) as not_comp, size(nodes) as
288 len_nodes
289 return i, pull_requests, match_relationships, nodes,
290 relationships, toFloat(not_comp) / toFloat(len_nodes
291 ) as proportion

```

Listing 1: Competing PRs Workflow Type Query

```

289 call {

```

```

291 with [43529, 92173, 56349, 56350, 56355, 53854, 53855,
292 53860, 53861, 61030, 57447, 61029, 61035, 68741,
293 68742, 90257, 68768, 68768, 15019, 6833, 3252, 3253, 57529,
294 6842, 57530, 90300, 90299, 81091, 33989, 33990,
295 33991, 57552, 87767, 87768, 87784, 6892, 6893,
296 34030, 34031, 69361, 69362, 69363, 69366, 54016,
297 54017, 54018, 34058, 3347, 6950, 90932, 90933,
298 90953, 81229, 81230, 86349, 81234, 43353, 36698,
299 36699, 43354, 86361, 36703, 60257, 60258, 60259,
300 69990, 69991, 70008, 39816, 39817, 39818, 83356,
301 83357, 83358, 28581, 28583, 80808, 80814, 86446,
302 29104, 29105, 29106, 15802, 15806, 87487, 87496,
303 87497, 29132, 40420, 40421, 40422, 9199, 92150,
304 92151, 80890, 57342, 57343] as all_ids // see
305 cypher_scripts/fetch_all_pr_stack_ids
306 match (hub)-[r]-(pr2:pull_request {status: "merged"})
307 where ((hub:pull_request and hub.status = "merged") or (
308 hub:issue and hub.status = "closed")) and pr2.
309 creation_date < hub.creation_date and not id(hub) in
310 all_ids and not id(pr2) in all_ids
311 with hub, collect(distinct pr2) as prs, collect(distinct
312 pr2.user) as users, collect(distinct id(pr2)) as
313 pr_ids
314 where size(prs) >= 3 and size(users) >= 2
315 return collect(distinct id(hub))+apoc.coll.toSet(apoc.
316 coll.flatten(collect(pr_ids))) as known_hubs
317 }
318
319 call {
320 with known_hubs
321 with known_hubs+[80808, 92085] as exclude_ids
322 match (i:issue {status:"closed"})-[r {labels: "fixes"}]-
323 (p:pull_request {status: "merged"}), (i2:issue {
324 status:"closed"})-[r2]-(p), (i2)-[r3]-(p2:
325 pull_request {status:"merged"})
326 where (i2.creation_date > p.creation_date or i2.
327 creation_date > i.creation_date) and i.number <> i2.
328 number and p.number <> p2.number and not id(p) in
329 exclude_ids and not id(p2) in exclude_ids
330 return collect(distinct id(i))+collect(distinct id(p))+
331 collect(distinct id(i2))+collect(distinct id(p2)) as
332 known_consq_2
333 }
334
335 call {
336 match (i:issue {status:"closed"})-[r {labels: "fixes"}]-
337 (p:pull_request {status: "merged"}), (i2:issue)-[r2
338 ]-(p)
339 where i2.creation_date > p.creation_date and i.number <>
340 i2.number
341 return collect(distinct id(i)) as known_consq
342 }
343
344 with known_hubs+known_consq_2 as exclude_ids ,
345 known_consq
346 match (i:issue {status:"closed"})-[r {labels: "fixes"}]-
347 (p:pull_request {status: "merged"}), (i2:issue)-[r2
348 ]-(p)
349 where i2.creation_date > p.creation_date and i.number <>
350 i2.number and not id(i) in exclude_ids and not id(p)
351 in exclude_ids and not id(i2) in exclude_ids
352 with i, p, i2, [r,r2] as match_relationships, known_consq
353 call apoc.path.subgraphAll(i, {limit: 50, bfs: true })
354 yield nodes, relationships
355 with i, p, i2, nodes, relationships, match_relationships,
356 size(collect([i_node in nodes where i_node.type="
357 issue" and i_node.status="closed" and i_node.number
358 <> i.number and not id(i_node) in known_consq])) as
359 not_consq, size(nodes) as len_nodes
360 return i, p, i2, nodes, relationships,
361 match_relationships,toFloat(not_consq) / toFloat(
362 len_nodes) as proportion

```

Listing 2: Consequent Issue Workflow Type Query

```

349 call {
350 with [43529, 92173, 56349, 56350, 56355, 53854, 53855,
351 53860, 53861, 61030, 57447, 61029, 61035, 68741,
352 68742, 90257, 68768, 15019, 6833, 3252, 3253, 57529,
353 6842, 57530, 90300, 90299, 81091, 33989, 33990,
354 33991, 57552, 87767, 87768, 87784, 6892, 6893,
355 34030, 34031, 69361, 69362, 69363, 69366, 54016,
356 54017, 54018, 34058, 3347, 6950, 90932, 90933,
357 90953, 81229, 81230, 86349, 81234, 43353, 36698,
358 36699, 43354, 86361, 36703, 60257, 60258, 60259,
359 69990, 69991, 70008, 39816, 39817, 39818, 83356,
360 83357, 83358, 28581, 28583, 80808, 80814, 86446,
361 29104, 29105, 29106, 15802, 15806, 87487, 87496,
362 87497, 29132, 40420, 40421, 40422, 9199, 92150,
363 92151, 80890, 57342, 57343] as all_ids // see
364 cypher_scripts/fetch_all_pr_stack_ids
365 match (hub)-[r]-(pr2:pull_request {status: "merged"})
366 where (hub:pull_request and hub.status = "merged") or (
367 hub:issue and hub.status = "closed") and pr2.
368 creation_date < hub.creation_date and not id(hub) in
369 all_ids and not id(pr2) in all_ids
370 with hub, collect(distinct pr2) as prs, collect(distinct
371 pr2.user) as users, collect(distinct id(pr2)) as
372 pr_ids
373 where size(prs) >= 3 and size(users) >= 2
374 return collect(distinct id(hub))+apoc.coll.toSet(apoc.
375 coll.flatten(collect(pr_ids))) as known_hubs
376 }
377
378 call {
379 with known_hubs
380 with known_hubs+[80808, 92085] as exclude_ids
381 match (i:issue {status:"closed"})-[r {labels: "fixes"}]-
382 (p:pull_request {status: "merged"}), (i2:issue {
383 status:"closed"})-[r2]-(p), (i2)-[r3]-(p2:
384 pull_request {status:"merged"})
385 where (i2.creation_date > p.creation_date or i2.
386 creation_date > i.creation_date) and i.number <> i2.
387 number and p.number <> p2.number and not id(p) in
388 exclude_ids and not id(p2) in exclude_ids and not id
389 (i) in exclude_ids and not id(i2) in exclude_ids
390 with i, p, i2, p2, [r,r2,r3] as match_relationships,
391 known_consq
392 call apoc.path.subgraphAll(i, {limit: 50, bfs: true })
393 yield nodes, relationships
394 with i, p, i2, p2, nodes, relationships,
395 match_relationships, size(collect([i_node in nodes
396 where i_node.type="issue" and i_node.status="closed"
397 and i_node.number <> i.number and i_node.number <>
398 i2.number and not id(i_node) in known_consq])) as
399 not_consq, size(nodes) as len_nodes
400 return i, p, i2, p2, nodes, relationships,
401 match_relationships, toFloat(not_consq) / toFloat(
402 len_nodes) as proportion
403 }
404
405 with known_hubs+[80808, 92085] as exclude_ids,
406 known_consq
407 match (i:issue {status:"closed"})-[r {labels: "fixes"}]-
408 (p:pull_request {status: "merged"}), (i2:issue {
409 status:"closed"})-[r2]-(p), (i2)-[r3]-(p2:
410 pull_request {status:"merged"})
411 where (i2.creation_date > p.creation_date or i2.
412 creation_date > i.creation_date) and i.number <> i2.
413 number and p.number <> p2.number and not id(p) in
414 exclude_ids and not id(p2) in exclude_ids and not id
415 (i) in exclude_ids and not id(i2) in exclude_ids
416 with i, p, i2, p2, [r,r2,r3] as match_relationships,
417 known_consq
418 call apoc.path.subgraphAll(i, {limit: 50, bfs: true })
419 yield nodes, relationships
420 with i, p, i2, p2, nodes, relationships,
421 match_relationships, size(collect([i_node in nodes
422 where i_node.type="issue" and i_node.status="closed"
423 and i_node.number <> i.number and i_node.number <>
424 i2.number and not id(i_node) in known_consq])) as
425 not_consq, size(nodes) as len_nodes
426 return i, p, i2, p2, nodes, relationships,
427 match_relationships, toFloat(not_consq) / toFloat(
428 len_nodes) as proportion
429 }

```

Listing 3: Consequent Issue-PR Workflow Type Query

```

407 where not (pr)--(pr_2) and pr.number <> pr_2.number and
408 ((pr.status = "closed" and (pr)--(pr_2) and pr_2.
409 status="merged") or pr.status <> "closed")
410 with i, collect(distinct pr)+collect(distinct pr_2) as
411 pull_requests, collect(distinct r)+collect(distinct
412 r_2) as match_relationships, collect(distinct id(pr))
413 +collect(distinct id(pr_2)) as known_decomposition
414 where size(pull_requests) > 1 and size([p in
415 pull_requests where p.status = "merged"]) >= toFloat
416 (size(pull_requests))/ toFloat(2)
417 return apoc.coll.toSet(apoc.coll.flatten(collect(
418 known_decomposition))) as known_decomposition
419 }
420
421 match (i:issue {status: "closed"})-[r {labels: "fixes
422 "}]-(pr:pull_request), (i)-[r2 {labels: "fixes"}]-
423 (pr_2:pull_request)
424 where not (pr)--(pr_2) and pr.number <> pr_2.number and
425 ((pr.status = "closed" and (pr)--(pr_2) and pr_2.
426 status="merged") or pr.status <> "closed")
427 with i, collect(distinct pr)+collect(distinct pr_2) as
428 pull_requests, collect(distinct r)+collect(distinct
429 r_2) as match_relationships, known_decomposition
430 where size(pull_requests) > 1 and size([p in
431 pull_requests where p.status = "merged"]) >= toFloat
432 (size(pull_requests))/ toFloat(2)
433 call apoc.path.subgraphAll(i, {limit: case 50 > size(
434 pull_requests) when true then 50 when false then
435 size(pull_requests) + 1 end, bfs: true })
436 yield nodes, relationships
437 with i, pull_requests, nodes, relationships,
438 match_relationships, size(collect([i_node in nodes
439 where i_node.type="issue" and i_node.status="closed"
440 and i_node.number <> i.number and not id(i_node) in
441 known_decomposition])) as not_decomp, size(nodes)
442 as len_nodes
443 return i, pull_requests, nodes, relationships,
444 match_relationships, toFloat(not_decomp) / toFloat(
445 len_nodes) as proportion
446 }

```

Listing 4: Decomposed Issue Workflow Type Query

```

441 with [43529, 92173, 56349, 56350, 56355, 53854, 53855,
442 53860, 53861, 61030, 57447, 61029, 61035, 68741,
443 68742, 90257, 68768, 15019, 6833, 3252, 3253, 57529,
444 6842, 57530, 90300, 90299, 81091, 33989, 33990,
445 33991, 57552, 87767, 87768, 87784, 6892, 6893,
446 34030, 34031, 69361, 69362, 69363, 69366, 54016,
447 54017, 54018, 34058, 3347, 6950, 90932, 90933,
448 90953, 81229, 81230, 86349, 81234, 43353, 36698,
449 36699, 43354, 86361, 36703, 60257, 60258, 60259,
450 69990, 69991, 70008, 39816, 39817, 39818, 83356,
451 83357, 83358, 28581, 28583, 80808, 80814, 86446,
452 29104, 29105, 29106, 15802, 15806, 87487, 87496,
453 87497, 29132, 40420, 40421, 40422, 9199, 92150,
454 92151, 80890, 57342, 57343] as known_stacks // see
455 cypher_scripts/fetch_all_pr_stack_ids
456 match (pr:pull_request)-[r1]->(pr_2:pull_request)-[r2]->(
457 pr_3:pull_request)
458 where pr.creation_date < pr_2.creation_date < pr_3.
459 creation_date
460 optional match (optional_issue:issue)-[optional_r]-(pr)
461 unwind [pr.number, pr_2.number, pr_3.number] as
462 number_list
463 unwind [pr.status, pr_2.status, pr_3.status] as
464 status_list
465 unwind [pr.user, pr_2.user, pr_3.user] as user_list
466 with pr, pr_2, pr_3, collect(distinct number_list) as
467 number_list, [r1,r2] as match_relationships,
468 optional_issue, optional_r, status_list,user_list,
469 count(user_list) as user_counts, known_stacks
470 where optional_issue = null or (not (optional_issue)--
471 (pr_2) and not (optional_issue)--(pr_3))

```

```

465 with pr, pr_2, pr_3, number_list, match_relationships,
466 status_list, optional_issue, optional_r, apoc.agg.
467 maxItems(user_list, user_counts) as max_users,
468 known_stacks
469 where size(number_list) = 3 and not (:pull_request)-->(pr
470 ) and not (pr_3)-->(:pull_request) and size([i in
471 status_list where i = "merged"]) >= 3/2 and
472 max_users.value >= 3/2
473 call apoc.path.subgraphAll(pr, {limit: 50, bfs:true})
474 yield nodes, relationships
475 with pr, pr_2, pr_3, [id(pr), id(pr_2), id(pr_3)] as
476 all_ids, optional_issue, optional_r, size(collect([
477 i_node in nodes where i_node.type="pull_request" and
478 i_node.number <> pr.number and not id(i_node) in
479 known_stacks])) as not_stack, size(nodes) as
480 len_nodes, nodes, relationships, match_relationships
481 return pr, pr_2, pr_3, all_ids, optional_issue,
482 optional_r, nodes, relationships, toFloat(not_stack)
483 / toFloat(len_nodes) as proportion,
484 match_relationships

```

Listing 5: Dependent PRs Workflow Type Query

```

484 call {
485 match (pr:pull_request {status: "merged"})-[r {labels:"
486 fixes"}]-(i:issue)
487 with pr, collect(distinct i) as issues, collect(distinct
488 r) as match_relationships
489 where size([issue in issues where issue.status="closed"])
490 > 1
491 return collect(distinct id(pr)) as known_dependent
492 }
493 match (pr:pull_request {status: "merged"})-[r {labels:"
494 fixes"}]-(i:issue)
495 with pr, collect(distinct i) as issues, collect(distinct
496 r) as match_relationships, known_dependent
497 where size([issue in issues where issue.status="closed"])
498 > 1
499 with pr, [issue in issues where issue.status="closed"] as
500 closed_issues, match_relationships, known_dependent
501 call apoc.path.subgraphAll(pr, {limit: case 50 > size(
502 closed_issues) + 1 when true then 50 when false then
503 size(closed_issues) + 1 end, bfs: true })
504 yield nodes, relationships
505 with pr, closed_issues, nodes, relationships,
506 match_relationships, size(collect([i_node in nodes
507 where i_node.type="pull_request" and i_node.status="
508 merged" and i_node.number <> pr.number and not id(
509 i_node) in known_dependent])) as not_dep, size(nodes
510 ) as len_nodes
511 return pr, closed_issues, nodes, relationships,
512 match_relationships, toFloat(not_dep) / toFloat(
513 len_nodes) as proportion

```

Listing 6: Divergent PR Workflow Type Query

```

511 call {
512 match (i:issue)-[r {labels:"duplicate"}]-(i2:issue)
513 where i2.creation_date > i.creation_date and i2.user <> i
514 .user
515 with i, collect(distinct i2) as spoke_issues, collect(
516 distinct r) as match_relationships, collect(distinct
517 id(i)) as known_dups
518 where size(spoke_issues) > 1
519 return apoc.coll.toSet(apoc.coll.flatten(collect(
520 known_dups))) as known_dups
521 }
522 match (i:issue)-[r {labels:"duplicate"}]-(i2:issue)
523 where i2.creation_date > i.creation_date and i2.user <> i
524 .user

```

```

523 with i, collect(distinct i2) as spoke_issues, collect(
524 distinct r) as match_relationships, known_dups
525 where size(spoke_issues) > 1
526 call apoc.path.subgraphAll(i, {limit: case 50 > size(
527 spoke_issues) when true then 50 when false then size
528 (spoke_issues) + 1 end, bfs: true })
529 yield nodes, relationships
530 with i, spoke_issues, nodes, relationships,
531 match_relationships, size(collect([i_node in nodes
532 where i_node.type="issue" and i_node.number <> i.
533 number and not id(i_node) in known_dups])) as
534 not_dups, size(nodes) as len_nodes
535 return i, spoke_issues, nodes, relationships,
536 match_relationships, toFloat(not_dups) / toFloat(
537 len_nodes) as proportion

```

Listing 7: Duplicate Issue Hub Workflow Type Query

```

540 call {
541 with [43529, 92173, 56349, 56350, 56355, 53854, 53855,
542 53860, 53861, 61030, 57447, 61029, 61035, 68741,
543 68742, 90257, 68768, 15019, 6833, 3252, 3253, 57529,
544 6842, 57530, 90300, 90299, 81091, 33989, 33990,
545 33991, 57552, 87767, 87768, 87784, 6892, 6893,
546 34030, 34031, 69361, 69362, 69363, 69366, 54016,
547 54017, 54018, 34058, 3347, 6950, 90932, 90933,
548 90953, 81229, 81230, 86349, 81234, 43353, 36698,
549 36699, 43354, 86361, 36703, 60257, 60258, 60259,
550 69990, 69991, 70008, 39816, 39817, 39818, 83356,
551 83357, 83358, 28581, 28583, 80808, 80814, 86446,
552 29104, 29105, 29106, 15802, 15806, 87487, 87496,
553 87497, 29132, 40420, 40421, 40422, 9199, 92150,
554 92151, 80890, 57342, 57343] as all_ids // see
555 cypher_scripts/fetch_all_pr_stack_ids
556 match (hub)-[r]-(pr2:pull_request {status: "merged"})
557 where ((hub:pull_request and hub.status = "merged") or (
558 hub:issue and hub.status = "closed")) and pr2.
559 creation_date < hub.creation_date and not id(hub) in
560 all_ids and not id(pr2) in all_ids
561 with hub, collect(distinct pr2) as prs, collect(distinct
562 pr2.user) as users, collect(distinct id(pr2)) as
563 pr_ids
564 where size(prs) >= 3 and size(users) >= 2
565 return collect(distinct id(hub))+apoc.coll.toSet(apoc.
566 coll.flatten(collect(pr_ids))) as all_ids
567 }
568 call {
569 with all_ids
570 match (pr1:pull_request)-[r {labels:"fixes"}]-(i:issue),(
571 pr1)-[r2]-(pr2:pull_request),(pr2)-[r3 {labels:"
572 fixes"}]-(i)
573 where not id(i) in all_ids and not id(pr1) in all_ids and
574 not id(pr2) in all_ids and pr1.number<>pr2.number
575 and i.status = "closed" and pr1.status="merged" and
576 pr2.status="merged" and pr2.creation_date > pr1.
577 creation_date
578 return collect(distinct id(i)) as known_extended
579 }
580 match (pr1:pull_request)-[r {labels:"fixes"}]-(i:issue),(
581 pr1)-[r2]-(pr2:pull_request),(pr2)-[r3 {labels:"
582 fixes"}]-(i)
583 where not id(i) in all_ids and not id(pr1) in all_ids and
584 not id(pr2) in all_ids and pr1.number<>pr2.number
585 and i.status = "closed" and pr1.status="merged" and
586 pr2.status="merged" and pr2.creation_date > pr1.
587 creation_date
588 with i, pr1, pr2, [r,r2,r3] as match_relationships,
589 known_extended
590 call apoc.path.subgraphAll(i, {limit: 50, bfs:true})
591 yield nodes, relationships

```



```

581 with i, pr1, pr2, nodes, relationships,
582     match_relationships, size(collect([i_node in nodes
583     where i_node.type="issue" and i_node.status="closed"
584     and i_node.number <> i.number and not id(i_node) in
585     known_extended])) as not_ext, size(nodes) as
586     len_nodes
587 return i, pr1, pr2, nodes, relationships,
588     match_relationships, toFloat(not_ext) / toFloat(
589     len_nodes) as proportion

```

Listing 8: Extended PR Workflow Type Query

```

639 with hub, prs, nodes, relationships, match_relationships,
640     size(collect([i_node in nodes where (i_node.type="
641     pull_request" and hub:pull_request and i_node.status
642     = "merged") or (i_node.type="issue" and hub:issue
643     and i_node.status = "closed") and i_node.number <>
644     hub.number and not id(i_node) in known_hubs])) as
645     not_prhub, size(nodes) as len_nodes
646 return hub, prs, nodes, relationships,
647     match_relationships, toFloat(not_prhub) / toFloat(
648     len_nodes) as proportion

```

Listing 9: Integrating PR/Issue Hub Workflow Type Query

```

595 call {
596 with [43529, 92173, 56349, 56350, 56355, 53854, 53855,
597     53860, 53861, 61030, 57447, 61029, 61035, 68741,
598     68742, 90257, 68768, 15019, 6833, 3252, 3253, 57529,
599     6842, 57530, 90300, 90299, 81091, 33989, 33990,
600     33991, 57552, 87767, 87768, 87784, 6892, 6893,
601     34030, 34031, 69361, 69362, 69363, 69366, 54016,
602     54017, 54018, 34058, 3347, 6950, 90932, 90933,
603     90953, 81229, 81230, 86349, 81234, 43353, 36698,
604     36699, 43354, 86361, 36703, 60257, 60258, 60259,
605     69990, 69991, 70008, 39816, 39817, 39818, 83356,
606     83357, 83358, 28581, 28583, 80808, 80814, 86446,
607     29104, 29105, 29106, 15802, 15806, 87487, 87496,
608     87497, 29132, 40420, 40421, 40422, 9199, 92150,
609     92151, 80890, 57342, 57343] as all_ids // see
610     cypher_scripts/fetch_all_pr_stack_ids
611 match (hub)-[r]-(pr2:pull_request {status: "merged"})
612 where ((hub:pull_request and hub.status = "merged") or (
613     hub:issue and hub.status = "closed")) and pr2.
614     creation_date < hub.creation_date and not id(hub) in
615     all_ids and not id(pr2) in all_ids
616 with hub, collect(distinct pr2) as prs, collect(distinct
617     pr2.user) as users, collect(distinct id(pr2)) as
618     pr_ids
619 where size(prs) >= 3 and size(users) >= 2
620 return collect(distinct id(hub))+apoc.coll.toSet(apoc.
621     coll.flatten(collect(pr_ids))) as known_hubs
622 }
623
624 with [43529, 92173, 56349, 56350, 56355, 53854, 53855,
625     53860, 53861, 61030, 57447, 61029, 61035, 68741,
626     68742, 90257, 68768, 15019, 6833, 3252, 3253, 57529,
627     6842, 57530, 90300, 90299, 81091, 33989, 33990,
628     33991, 57552, 87767, 87768, 87784, 6892, 6893,
629     34030, 34031, 69361, 69362, 69363, 69366, 54016,
630     54017, 54018, 34058, 3347, 6950, 90932, 90933,
631     90953, 81229, 81230, 86349, 81234, 43353, 36698,
632     36699, 43354, 86361, 36703, 60257, 60258, 60259,
633     69990, 69991, 70008, 39816, 39817, 39818, 83356,
634     83357, 83358, 28581, 28583, 80808, 80814, 86446,
635     29104, 29105, 29106, 15802, 15806, 87487, 87496,
636     87497, 29132, 40420, 40421, 40422, 9199, 92150,
637     92151, 80890, 57342, 57343] as all_ids, known_hubs
638 // see cypher_scripts/fetch_all_pr_stack_ids
639 match (hub)-[r]-(pr2:pull_request {status: "merged"})
640 where ((hub:pull_request and hub.status = "merged") or (
641     hub:issue and hub.status = "closed")) and pr2.
642     creation_date < hub.creation_date and not id(hub) in
643     all_ids and not id(pr2) in all_ids
644 with hub, collect(distinct pr2) as prs, collect(distinct
645     pr2.user) as users, collect(distinct r) as
646     match_relationships, known_hubs
647 where size(prs) >= 3 and size(users) >= 2
648 with hub, prs, match_relationships, known_hubs
649 call apoc.path.subgraphAll(hub, {limit: case 50 > size(
650     prs) when true then 50 when false then size(prs) + 1
651     end, bfs: true })
652 yield nodes, relationships

```