# TECHNICAL REPORT

# SIMILARITY QUERIES - TRANSFORMATION RULES AND PROOFS

YASIN N. SILVA, Arizona State University, ysilva@asu.edu

WALID G. AREF, Purdue University, aref@cs.purdue.edu

PER-AKE LARSON, Microsoft Research, palarson@microsoft.com

SPENCER S. PEARSON, Arizona State University, sspearso@asu.edu

MOHAMED H. ALI, Microsoft Corporation, mali@microsoft.com

School of Mathematical and Natural Sciences

Arizona State University

September 10, 2012

## A. TRANSFORMATION RULES FOR SIMILARITY-AWARE OPERATORS

### Combining/Separating Similarity Selection Predicates

R1. $\sigma_{\theta_{\varepsilon 1,C1}(e) \cap \theta_{\varepsilon 2,C2}(e)}(E) \equiv \sigma_{\theta_{\varepsilon 1,C1}(e)}\left(\sigma_{\theta_{\varepsilon 2,C2}(e)}(E)\right)$.
R2. $\sigma_{\theta_{\varepsilon,C1}(e) \cap \theta_{kNN,C2}(e)}(E) \equiv \sigma_{\theta_{\varepsilon,C1}(e)}\left(\sigma_{\theta_{kNN,C2}(e)}(E)\right)$.
R3. $\sigma_{\theta_{kNN,C1}(e) \cap \theta_{\varepsilon,C2}(e)}(E) \not\equiv \sigma_{\theta_{kNN,C1}(e)}\left(\sigma_{\theta_{\varepsilon,C2}(e)}(E)\right)$.
R4. $\sigma_{\theta_{kNN1,C1}(e) \cap \theta_{kNN2,C2}(e)}(E) \not\equiv \sigma_{\theta_{kNN1,C1}(e)}\left(\sigma_{\theta_{kNN2,C2}(e)}(E)\right)$.

### Combining/Separating Similarity Join and Similarity Selection

When the selection predicate attribute is the inner attribute in the join predicate:

R5. $\sigma_{\theta_{\varepsilon 1}(e1,e2) \cap \theta_{\varepsilon 2,C}(e2)}(E) \equiv \sigma_{\theta_{\varepsilon 1}(e1,e2)}(\sigma_{\theta_{\varepsilon 2,C}(e2)}(E)) \equiv \sigma_{\theta_{\varepsilon 2,C}(e2)}(\sigma_{\theta_{\varepsilon 1}(e1,e2)}(E))$.
R6. $\sigma_{\theta_{\varepsilon}(e1,e2) \cap \theta_{kNN,C}(e2)}(E) \equiv \sigma_{\theta_{\varepsilon}(e1,e2)}(\sigma_{\theta_{kNN,C}(e2)}(E))$.
R7. $\sigma_{\theta_{\varepsilon}(e1,e2) \cap \theta_{kNN,C}(e2)}(E) \not\equiv \sigma_{\theta_{kNN,C}(e2)}(\sigma_{\theta_{\varepsilon}(e1,e2)}(E))$.
R8. $\sigma_{\theta_{kNN}(e1,e2) \cap \theta_{\varepsilon,C}(e2)}(E) \not\equiv \sigma_{\theta_{kNN}(e1,e2)}(\sigma_{\theta_{\varepsilon,C}(e2)}(E))$.
R9. $\sigma_{\theta_{kNN}(e1,e2) \cap \theta_{\varepsilon,C}(e2)}(E) \equiv \sigma_{\theta_{\varepsilon,C}(e2)}(\sigma_{\theta_{kNN}(e1,e2)}(E))$.
R10. $\sigma_{\theta_{kNN}(e1,e2) \cap \theta_{kNN,C}(e2)}(E) \not\equiv \sigma_{\theta_{kNN}(e1,e2)}(\sigma_{\theta_{kNN,C}(e2)}(E))$.
R11. $\sigma_{\theta_{kNN}(e1,e2) \cap \theta_{kNN,C}(e2)}(E) \not\equiv \sigma_{\theta_{kNN,C}(e2)}(\sigma_{\theta_{kNN}(e1,e2)}(E))$.
R12. $\sigma_{\theta_{kD}(e1,e2) \cap \theta_{\varepsilon,C}(e2)}(E) \not\equiv \sigma_{\theta_{kD}(e1,e2)}(\sigma_{\theta_{\varepsilon,C}(e2)}(E))$.
R13. $\sigma_{\theta_{kD}(e1,e2) \cap \theta_{\varepsilon,C}(e2)}(E) \equiv \sigma_{\theta_{\varepsilon,C}(e2)}(\sigma_{\theta_{kD}(e1,e2)}(E))$.
R14. $\sigma_{\theta_{kD}(e1,e2) \cap \theta_{kNN,C}(e2)}(E) \not\equiv \sigma_{\theta_{kD}(e1,e2)}(\sigma_{\theta_{kNN,C}(e2)}(E))$.
R15. $\sigma_{\theta_{kD}(e1,e2) \cap \theta_{kNN,C}(e2)}(E) \not\equiv \sigma_{\theta_{kNN,C}(e2)}(\sigma_{\theta_{kD}(e1,e2)}(E))$.
R16. $\sigma_{\theta_{A}(e1,e2) \cap \theta_{\varepsilon,C}(e2)}(E) \not\equiv \sigma_{\theta_{A}(e1,e2)}(\sigma_{\theta_{\varepsilon,C}(e2)}(E))$.
R17. $\sigma_{\theta_{A}(e1,e2) \cap \theta_{\varepsilon,C}(e2)}(E) \equiv \sigma_{\theta_{\varepsilon,C}(e2)}(\sigma_{\theta_{A}(e1,e2)}(E))$.
R18. $\sigma_{\theta_{A}(e1,e2) \cap \theta_{kNN,C}(e2)}(E) \not\equiv \sigma_{\theta_{A}(e1,e2)}(\sigma_{\theta_{kNN,C}(e2)}(E))$.
R19. $\sigma_{\theta_{A}(e1,e2) \cap \theta_{kNN,C}(e2)}(E) \not\equiv \sigma_{\theta_{kNN,C}(e2)}(\sigma_{\theta_{A}(e1,e2)}(E))$.

When the selection predicate attribute is the outer attribute in the join predicate:

R20. $\sigma_{\theta_{\varepsilon 1}(e1,e2) \cap \theta_{\varepsilon 2,C}(e1)}(E) \equiv \sigma_{\theta_{\varepsilon 1}(e1,e2)}(\sigma_{\theta_{\varepsilon 2,C}(e1)}(E)) \equiv \sigma_{\theta_{\varepsilon 2,C}(e1)}(\sigma_{\theta_{\varepsilon 1}(e1,e2)}(E))$.
R21. $\sigma_{\theta_{\varepsilon}(e1,e2) \cap \theta_{kNN,C}(e1)}(E) \equiv \sigma_{\theta_{\varepsilon}(e1,e2)}(\sigma_{\theta_{kNN,C}(e1)}(E))$.
R22. $\sigma_{\theta_{\varepsilon}(e1,e2) \cap \theta_{kNN,C}(e1)}(E) \not\equiv \sigma_{\theta_{kNN,C}(e1)}(\sigma_{\theta_{\varepsilon}(e1,e2)}(E))$.
R23. $\sigma_{\theta_{kNN}(e1,e2) \cap \theta_{\varepsilon,C}(e1)}(E) \equiv \sigma_{\theta_{kNN}(e1,e2)}(\sigma_{\theta_{\varepsilon,C}(e1)}(E)) \equiv \sigma_{\theta_{\varepsilon,C}(e1)}(\sigma_{\theta_{kNN}(e1,e2)}(E))$.
R24. $\sigma_{\theta_{kNN}(e1,e2) \cap \theta_{kNN,C}(e1)}(E) \equiv \sigma_{\theta_{kNN}(e1,e2)}(\sigma_{\theta_{kNN,C}(e1)}(E)) \equiv \sigma_{\theta_{kNN,C}(e1)}(\sigma_{\theta_{kNN}(e1,e2)}(E))$.
R25. $\sigma_{\theta_{kD}(e1,e2) \cap \theta_{\varepsilon,C}(e1)}(E) \not\equiv \sigma_{\theta_{kD}(e1,e2)}(\sigma_{\theta_{\varepsilon,C}(e1)}(E))$.
R26. $\sigma_{\theta_{kD}(e1,e2) \cap \theta_{\varepsilon,C}(e1)}(E) \equiv \sigma_{\theta_{\varepsilon,C}(e1)}(\sigma_{\theta_{kD}(e1,e2)}(E))$.
R27. $\sigma_{\theta_{kD}(e1,e2) \cap \theta_{kNN,C}(e1)}(E) \not\equiv \sigma_{\theta_{kD}(e1,e2)}(\sigma_{\theta_{kNN,C}(e1)}(E))$.
R28. $\sigma_{\theta_{kD}(e1,e2) \cap \theta_{kNN,C}(e1)}(E) \not\equiv \sigma_{\theta_{kNN,C}(e1)}(\sigma_{\theta_{kD}(e1,e2)}(E))$.
R29. $\sigma_{\theta_{A}(e1,e2) \cap \theta_{\varepsilon,C}(e1)}(E) \equiv \sigma_{\theta_{A}(e1,e2)}(\sigma_{\theta_{\varepsilon,C}(e1)}(E)) \equiv \sigma_{\theta_{\varepsilon,C}(e1)}(\sigma_{\theta_{A}(e1,e2)}(E))$.
R30. $\sigma_{\theta_{A}(e1,e2) \cap \theta_{kNN,C}(e1)}(E) \equiv \sigma_{\theta_{A}(e1,e2)}(\sigma_{\theta_{kNN,C}(e1)}(E))$.
R31. $\sigma_{\theta_{A}(e1,e2) \cap \theta_{kNN,C}(e1)}(E) \not\equiv \sigma_{\theta_{kNN,C}(e1)}(\sigma_{\theta_{A}(e1,e2)}(E))$.

### Combining/Separating Similarity Join Predicates

When the attributes in the predicates have a single direction (e1→e2, e2→e3):

R32. $\sigma_{\theta_{\varepsilon 1}(e1,e2) \cap \theta_{\varepsilon 2}(e2,e3)}(E) \equiv \sigma_{\theta_{\varepsilon 1}(e1,e2)}(\sigma_{\theta_{\varepsilon 2}(e2,e3)}(E)) \equiv \sigma_{\theta_{\varepsilon 2}(e2,e3)}(\sigma_{\theta_{\varepsilon 1}(e1,e2)}(E))$.
R33. $\sigma_{\theta_{kNN1}(e1,e2) \cap \theta_{kNN2}(e2,e3)}(E) \equiv \sigma_{\theta_{kNN1}(e1,e2)}(\sigma_{\theta_{kNN2}(e2,e3)}(E)) \equiv \sigma_{\theta_{kNN2}(e2,e3)}(\sigma_{\theta_{kNN1}(e1,e2)}(E))$.
R34. $\sigma_{\theta_{kD1}(e1,e2) \cap \theta_{kD2}(e2,e3)}(E) \not\equiv \sigma_{\theta_{kD1}(e1,e2)}(\sigma_{\theta_{kD2}(e2,e3)}(E))$.

R35. $\sigma_{\theta_{kD1}(e1,e2)\cap\theta_{kD2}(e2,e3)}(E) \not\equiv \sigma_{\theta_{kD2}(e2,e3)}(\sigma_{\theta_{kD1}(e1,e2)}(E))$.
R36. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{kNN}(e2,e3)}(E) \equiv \sigma_{\theta_{\varepsilon}(e1,e2)}(\sigma_{\theta_{kNN}(e2,e3)}(E)) \equiv \sigma_{\theta_{kNN}(e2,e3)}(\sigma_{\theta_{\varepsilon}(e1,e2)}(E))$.
R37. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{kD}(e2,e3)}(E) \equiv \sigma_{\theta_{\varepsilon}(e1,e2)}(\sigma_{\theta_{kD}(e2,e3)}(E))$.
R38. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{kD}(e2,e3)}(E) \not\equiv \sigma_{\theta_{kD}(e2,e3)}(\sigma_{\theta_{\varepsilon}(e1,e2)}(E))$.
R39. $\sigma_{\theta_{kNN}(e1,e2)\cap\theta_{kD}(e2,e3)}(E) \not\equiv \sigma_{\theta_{kNN}(e1,e2)}(\sigma_{\theta_{kD}(e2,e3)}(E))$.
R40. $\sigma_{\theta_{kNN}(e1,e2)\cap\theta_{kD}(e2,e3)}(E) \not\equiv \sigma_{\theta_{kD}(e2,e3)}(\sigma_{\theta_{kNN}(e1,e2)}(E))$.
R41. $\sigma_{\theta_{A1}(e1,e2)\cap\theta_{A2}(e2,e3)}(E) \equiv \sigma_{\theta_{A1}(e1,e2)}(\sigma_{\theta_{A2}(e2,e3)}(E)) \equiv \sigma_{\theta_{A2}(e2,e3)}(\sigma_{\theta_{A1}(e1,e2)}(E))$.
R42. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{A}(e2,e3)}(E) \equiv \sigma_{\theta_{\varepsilon}(e1,e2)}(\sigma_{\theta_{A}(e2,e3)}(E)) \equiv \sigma_{\theta_{A}(e2,e3)}(\sigma_{\theta_{\varepsilon}(e1,e2)}(E))$.
R43. $\sigma_{\theta_{A}(e1,e2)\cap\theta_{kNN}(e2,e3)}(E) \equiv \sigma_{\theta_{A}(e1,e2)}(\sigma_{\theta_{kNN}(e2,e3)}(E))$.
R44. $\sigma_{\theta_{A}(e1,e2)\cap\theta_{kNN}(e2,e3)}(E) \equiv \sigma_{\theta_{kNN}(e2,e3)}(\sigma_{\theta_{A}(e1,e2)}(E))$.
R45. $\sigma_{\theta_{A}(e1,e2)\cap\theta_{kD}(e2,e3)}(E) \equiv \sigma_{\theta_{A}(e1,e2)}(\sigma_{\theta_{kD}(e2,e3)}(E))$.
R46. $\sigma_{\theta_{A}(e1,e2)\cap\theta_{kD}(e2,e3)}(E) \not\equiv \sigma_{\theta_{kD}(e2,e3)}(\sigma_{\theta_{A}(e1,e2)}(E))$.

When the predicates' attributes do not have a single direction (e1→e2, e2 ←e3):

R47. $\sigma_{\theta_{\varepsilon1}(e1,e2)\cap\theta_{\varepsilon2}(e3,e2)}(E) \equiv \sigma_{\theta_{\varepsilon1}(e1,e2)}(\sigma_{\theta_{\varepsilon2}(e3,e2)}(E)) \equiv \sigma_{\theta_{\varepsilon2}(e3,e2)}(\sigma_{\theta_{\varepsilon1}(e1,e2)}(E))$.
R48. $\sigma_{\theta_{kNN1}(e1,e2)\cap\theta_{kNN2}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kNN1}(e1,e2)}(\sigma_{\theta_{kNN2}(e3,e2)}(E))$.
R49. $\sigma_{\theta_{kNN1}(e1,e2)\cap\theta_{kNN2}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kNN2}(e3,e2)}(\sigma_{\theta_{kNN1}(e1,e2)}(E))$.
R50. $\sigma_{\theta_{kD1}(e1,e2)\cap\theta_{kD2}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kD1}(e1,e2)}(\sigma_{\theta_{kD2}(e3,e2)}(E))$.
R51. $\sigma_{\theta_{kD1}(e1,e2)\cap\theta_{kD2}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kD2}(e3,e2)}(\sigma_{\theta_{kD1}(e1,e2)}(E))$.
R52. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{kNN}(e3,e2)}(E) \equiv \sigma_{\theta_{\varepsilon}(e1,e2)}(\sigma_{\theta_{kNN}(e3,e2)}(E))$.
R53. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{kNN}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kNN}(e3,e2)}(\sigma_{\theta_{\varepsilon}(e1,e2)}(E))$.
R54. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{kD}(e3,e2)}(E) \equiv \sigma_{\theta_{\varepsilon}(e1,e2)}(\sigma_{\theta_{kD}(e3,e2)}(E))$.
R55. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{kD}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kD}(e3,e2)}(\sigma_{\theta_{\varepsilon}(e1,e2)}(E))$.
R56. $\sigma_{\theta_{kNN}(e1,e2)\cap\theta_{kD}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kNN}(e1,e2)}(\sigma_{\theta_{kD}(e3,e2)}(E))$.
R57. $\sigma_{\theta_{kNN}(e1,e2)\cap\theta_{kD}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kD}(e3,e2)}(\sigma_{\theta_{kNN}(e1,e2)}(E))$.
R58. $\sigma_{\theta_{A1}(e1,e2)\cap\theta_{A2}(e3,e2)}(E) \not\equiv \sigma_{\theta_{A1}(e1,e2)}(\sigma_{\theta_{A2}(e3,e2)}(E))$.
R59. $\sigma_{\theta_{A1}(e1,e2)\cap\theta_{A2}(e3,e2)}(E) \not\equiv \sigma_{\theta_{A2}(e3,e2)}(\sigma_{\theta_{A1}(e1,e2)}(E))$.
R60. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{A}(e3,e2)}(E) \equiv \sigma_{\theta_{\varepsilon}(e1,e2)}(\sigma_{\theta_{A}(e3,e2)}(E))$.
R61. $\sigma_{\theta_{\varepsilon}(e1,e2)\cap\theta_{A}(e3,e2)}(E) \not\equiv \sigma_{\theta_{A}(e3,e2)}(\sigma_{\theta_{\varepsilon}(e1,e2)}(E))$.
R62. $\sigma_{\theta_{A}(e1,e2)\cap\theta_{kNN}(e3,e2)}(E) \not\equiv \sigma_{\theta_{A}(e1,e2)}(\sigma_{\theta_{kNN}(e3,e2)}(E))$.
R63. $\sigma_{\theta_{A}(e1,e2)\cap\theta_{kNN}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kNN}(e3,e2)}(\sigma_{\theta_{A}(e1,e2)}(E))$.
R64. $\sigma_{\theta_{A}(e1,e2)\cap\theta_{kD}(e3,e2)}(E) \not\equiv \sigma_{\theta_{A}(e1,e2)}(\sigma_{\theta_{kD}(e3,e2)}(E))$.
R65. $\sigma_{\theta_{A}(e1,e2)\cap\theta_{kD}(e3,e2)}(E) \not\equiv \sigma_{\theta_{kD}(e3,e2)}(\sigma_{\theta_{A}(e1,e2)}(E))$.

### Commutativity of Similarity Join Operators

R66. $E \bowtie_{\theta_{\varepsilon}(e,f)} F \equiv E \bowtie_{\theta_{\varepsilon}(f,e)} F$.
R67. $E \bowtie_{\theta_{kD}(e,f)} F \equiv E \bowtie_{\theta_{kD}(f,e)} F$.
R68. $E \bowtie_{\theta_{kNN}(e,f)} F \not\equiv E \bowtie_{\theta_{kNN}(f,e)} F$.
R69. $E \bowtie_{\theta_{A}(e,f)} F \not\equiv E \bowtie_{\theta_{A}(f,e)} F$.

### Distribution of Selection over Similarity Join

When all the attributes of the selection predicate $\theta$ involve only the attributes of one of the relations being joined:

R70. $\sigma_{\theta(e)}\left(E \bowtie_{\theta_{\varepsilon}(e,f)} F\right) \equiv (\sigma_{\theta(e)}(E)) \bowtie_{\theta_{\varepsilon}(e,f)} F$.
R71. $\sigma_{\theta(f)}\left(E \bowtie_{\theta_{\varepsilon}(e,f)} F\right) \equiv E \bowtie_{\theta_{\varepsilon}(e,f)} (\sigma_{\theta(f)}(F))$.
R72. $\sigma_{\theta(e)}\left(E \bowtie_{\theta_{kNN}(e,f)} F\right) \equiv (\sigma_{\theta(e)}(E)) \bowtie_{\theta_{kNN}(e,f)} F$.
R73. $\sigma_{\theta(f)}\left(E \bowtie_{\theta_{kNN}(e,f)} F\right) \not\equiv E \bowtie_{\theta_{kNN}(e,f)} (\sigma_{\theta(f)}(F))$.
R74. $\sigma_{\theta(e)}\left(E \bowtie_{\theta_{kD}(e,f)} F\right) \not\equiv (\sigma_{\theta(e)}(E)) \bowtie_{\theta_{kD}(e,f)} F$.

R75. $\sigma_{\theta(f)}\big(E \bowtie_{\theta_{kD}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{kD}(e,f)} (\sigma_{\theta(f)}(F))$.
R76. $\sigma_{\theta(e)}\big(E \bowtie_{\theta_{A}(e,f)} F\big) \equiv (\sigma_{\theta(e)}(E)) \bowtie_{\theta_{A}(e,f)} F$.
R77. $\sigma_{\theta(f)}\big(E \bowtie_{\theta_{A}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{A}(e,f)} (\sigma_{\theta(f)}(F))$.

When predicates $\theta_1$ and $\theta_2$ involve only the attributes of $E$ and $F$, respectively:

R78. $\sigma_{\theta_1(e)\wedge\theta_2(f)}\big(E \bowtie_{\theta_{\varepsilon}(e,f)} F\big) \equiv (\sigma_{\theta_1(e)}(E)) \bowtie_{\theta_{\varepsilon}(e,f)} (\sigma_{\theta_2(f)}(F))$.
R79. $\sigma_{\theta_1(e)\wedge\theta_2(f)}\big(E \bowtie_{\theta_{kNN}(e,f)} F\big) \not\equiv (\sigma_{\theta_1(e)}(E)) \bowtie_{\theta_{kNN}(e,f)} (\sigma_{\theta_2(f)}(F))$.
R80. $\sigma_{\theta_1(e)\wedge\theta_2(f)}\big(E \bowtie_{\theta_{kD}(e,f)} F\big) \not\equiv (\sigma_{\theta_1(e)}(E)) \bowtie_{\theta_{kD}(e,f)} (\sigma_{\theta_2(f)}(F))$.
R81. $\sigma_{\theta_1(e)\wedge\theta_2(f)}\big(E \bowtie_{\theta_{A}(e,f)} F\big) \not\equiv (\sigma_{\theta_1(e)}(E)) \bowtie_{\theta_{A}(e,f)} (\sigma_{\theta_2(f)}(F))$.

**Distribution of Similarity Selection over Join**

R82. $\sigma_{\theta_{\varepsilon,C}(e)}\big(E \bowtie_{\theta(e,f)} F\big) \equiv (\sigma_{\theta_{\varepsilon,C}(e)}(E)) \bowtie_{\theta(e,f)} F$.
R83. $\sigma_{\theta_{\varepsilon,C}(f)}\big(E \bowtie_{\theta(e,f)} F\big) \equiv E \bowtie_{\theta(e,f)} (\sigma_{\theta_{\varepsilon,C}(f)}(E))$.
R84. $\sigma_{\theta_{kNN,C}(e)}\big(E \bowtie_{\theta(e,f)} F\big) \not\equiv (\sigma_{\theta_{kNN,C}(e)}(E)) \bowtie_{\theta(e,f)} F$.
R85. $\sigma_{\theta_{kNN,C}(f)}\big(E \bowtie_{\theta(e,f)} F\big) \not\equiv E \bowtie_{\theta(e,f)} (\sigma_{\theta_{kNN,C}(f)}(E))$.

**Distribution of Similarity Selection over Similarity Join**

R86. $\sigma_{\theta_{\varepsilon 1,C}(e)}\big(E \bowtie_{\theta_{\varepsilon 2}(e,f)} F\big) \equiv (\sigma_{\theta_{\varepsilon 1,C}(e)}(E)) \bowtie_{\theta_{\varepsilon 2}(e,f)} F$.
R87. $\sigma_{\theta_{\varepsilon 1,C}(f)}\big(E \bowtie_{\theta_{\varepsilon 2}(e,f)} F\big) \equiv E \bowtie_{\theta_{\varepsilon 2}(e,f)} (\sigma_{\theta_{\varepsilon 1,C}(f)}(F))$.
R88. $\sigma_{\theta_{\varepsilon,C}(e)}\big(E \bowtie_{\theta_{kNN}(e,f)} F\big) \equiv (\sigma_{\theta_{\varepsilon,C}(e)}(E)) \bowtie_{\theta_{kNN}(e,f)} F$.
R89. $\sigma_{\theta_{\varepsilon,C}(f)}\big(E \bowtie_{\theta_{kNN}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{kNN}(e,f)} (\sigma_{\theta_{\varepsilon,C}(f)}(F))$.
R90. $\sigma_{\theta_{\varepsilon,C}(e)}\big(E \bowtie_{\theta_{kD}(e,f)} F\big) \not\equiv (\sigma_{\theta_{\varepsilon,C}(e)}(E)) \bowtie_{\theta_{kD}(e,f)} F$.
R91. $\sigma_{\theta_{\varepsilon,C}(f)}\big(E \bowtie_{\theta_{kD}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{kD}(e,f)} (\sigma_{\theta_{\varepsilon,C}(f)}(F))$.
R92. $\sigma_{\theta_{kNN,C}(e)}\big(E \bowtie_{\theta_{\varepsilon}(e,f)} F\big) \not\equiv (\sigma_{\theta_{kNN,C}(e)}(E)) \bowtie_{\theta_{\varepsilon}(e,f)} F$.
R93. $\sigma_{\theta_{kNN,C}(f)}\big(E \bowtie_{\theta_{\varepsilon}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{\varepsilon}(e,f)} (\sigma_{\theta_{kNN,C}(f)}(F))$.
R94. $\sigma_{\theta_{kNN1,C}(e)}\big(E \bowtie_{\theta_{kNN2}(e,f)} F\big) \equiv (\sigma_{\theta_{kNN1,C}(e)}(E)) \bowtie_{\theta_{kNN2}(e,f)} F$.
R95. $\sigma_{\theta_{kNN1,C}(f)}\big(E \bowtie_{\theta_{kNN2}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{kNN2}(e,f)} (\sigma_{\theta_{kNN1,C}(f)}(F))$.
R96. $\sigma_{\theta_{kNN,C}(e)}\big(E \bowtie_{\theta_{kD}(e,f)} F\big) \not\equiv (\sigma_{\theta_{kNN,C}(e)}(E)) \bowtie_{\theta_{kD}(e,f)} F$.
R97. $\sigma_{\theta_{kNN,C}(f)}\big(E \bowtie_{\theta_{kD}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{kD}(e,f)} (\sigma_{\theta_{kNN,C}(f)}(F))$.
R98. $\sigma_{\theta_{\varepsilon,C}(e)}\big(E \bowtie_{\theta_{A}(e,f)} F\big) \equiv (\sigma_{\theta_{\varepsilon,C}(e)}(E)) \bowtie_{\theta_{A}(e,f)} F$.
R99. $\sigma_{\theta_{\varepsilon,C}(f)}\big(E \bowtie_{\theta_{A}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{A}(e,f)} (\sigma_{\theta_{\varepsilon,C}(f)}(F))$.
R100. $\sigma_{\theta_{kNN,C}(e)}\big(E \bowtie_{\theta_{A}(e,f)} F\big) \not\equiv (\sigma_{\theta_{kNN,C}(e)}(E)) \bowtie_{\theta_{A}(e,f)} F$.
R101. $\sigma_{\theta_{kNN,C}(f)}\big(E \bowtie_{\theta_{A}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{A}(e,f)} (\sigma_{\theta_{kNN,C}(f)}(F))$.

**Associativity of Similarity Join Operators**

When the attributes in the predicates have a single direction (e→f, f→g):

R102. $\big(E \bowtie_{\theta_{\varepsilon 1}(e,f)} F\big) \bowtie_{\theta_{\varepsilon 2}(f,g)} G \equiv E \bowtie_{\theta_{\varepsilon 1}(e,f)} (F \bowtie_{\theta_{\varepsilon 2}(f,g)} G)$.
R103. $\big(E \bowtie_{\theta_{kNN1}(e,f)} F\big) \bowtie_{\theta_{kNN2}(f,g)} G \equiv E \bowtie_{\theta_{kNN1}(e,f)} (F \bowtie_{\theta_{kNN2}(f,g)} G)$.
R104. $\big(E \bowtie_{\theta_{kD1}(e,f)} F\big) \bowtie_{\theta_{kD2}(f,g)} G \not\equiv E \bowtie_{\theta_{kD1}(e,f)} (F \bowtie_{\theta_{kD2}(f,g)} G)$.
R105. $\big(E \bowtie_{\theta_{A1}(e,f)} F\big) \bowtie_{\theta_{A2}(f,g)} G \equiv E \bowtie_{\theta_{A1}(e,f)} (F \bowtie_{\theta_{A2}(f,g)} G)$.

When the predicates' attributes do not have a single direction (e→f, f←g):

R106. $G \bowtie_{\theta_{\varepsilon 1}(g,f)} \big(E \bowtie_{\theta_{\varepsilon 2}(e,f)} F\big) \equiv E \bowtie_{\theta_{\varepsilon 2}(e,f)} (G \bowtie_{\theta_{\varepsilon 1}(g,f)} F)$.
R107. $G \bowtie_{\theta_{kNN1}(g,f)} \big(E \bowtie_{\theta_{kNN2}(e,f)} F\big) \not\equiv E \bowtie_{\theta_{kNN2}(e,f)} (G \bowtie_{\theta_{kNN1}(g,f)} F)$.

R108. $G \bowtie_{\theta_{kD1}(g,f)} \left(E \bowtie_{\theta_{kD2}(e,f)} F\right) \not\equiv E \bowtie_{\theta_{kD2}(e,f)} (G \bowtie_{\theta_{kD1}(g,f)} F)$.
R109. $G \bowtie_{\theta_{A1}(g,f)} \left(E \bowtie_{\theta_{A2}(e,f)} F\right) \not\equiv E \bowtie_{\theta_{A2}(e,f)} (G \bowtie_{\theta_{A1}(g,f)} F)$.

### Applying Selection with a SJ predicate over Cross Product

R110. $\sigma_{\theta_{\varepsilon}(e,f)}(E \times F) \equiv E \bowtie_{\theta_{\varepsilon}(e,f)} F$.
R111. $\sigma_{\theta_{kNN}(e,f)}(E \times F) \equiv E \bowtie_{\theta_{kNN}(e,f)} F$.
R112. $\sigma_{\theta_{kD}(e,f)}(E \times F) \equiv E \bowtie_{\theta_{kD}(e,f)} F$.
R113. $\sigma_{\theta_{A}(e,f)}(E \times F) \equiv E \bowtie_{\theta_{A}(e,f)} F$.

### Rules that Take Advantage of Distance Function Properties

Pushing Selection Predicate under Originally Unrelated Ɛ-Join Operand.

R114. $\sigma_{\theta(e)}\left(E \bowtie_{\theta_{\varepsilon}(e,f)} F\right) \equiv (\sigma_{\theta(e)}(E)) \bowtie_{\theta_{\varepsilon}(e,f)} (\sigma_{\theta \pm \varepsilon(f)}(F))$.

Ɛ-Selection Predicate under Originally Unrelated Ɛ-Join Operand.

R115. $\sigma_{\theta_{\varepsilon 1,C}(e)}\left(E \bowtie_{\theta_{\varepsilon 2}(e,f)} F\right) \equiv (\sigma_{\theta_{\varepsilon 1,C}(e)}(E)) \bowtie_{\theta_{\varepsilon 2}(e,f)} (\sigma_{\theta_{(\varepsilon 1+\varepsilon 2),C}(f)}(F))$.

Associativity Rule that Enables Join on Originally Unrelated Attributes.

R116. $\left(E \bowtie_{\theta_{\varepsilon 1}(e,f)} F\right) \bowtie_{\theta_{\varepsilon 2}(f,g)} G \equiv \left(E \bowtie_{\theta_{\varepsilon 1+\varepsilon 2}(e,g)} G\right) \bowtie_{\theta_{\varepsilon 1}(e,f) \wedge \theta_{\varepsilon 2}(f,g)} F$.

### Eager and Lazy Transformations with SJ and SGB

Eager and Lazy Transformations with SGB and Join:

R117. The Eager and Lazy transformations can be extended to the case of SGB and regular join as shown in Theorem 1 (Section 4.4.1).

Eager and Lazy Transformations with Group-by and SJ:

R118. The Eager and Lazy aggregation transformations can be extended to the case of SJ and group-by as shown in Theorem 2 (Section 4.4.2).

Eager and Lazy Transformations with SGB and SJ:

R119. The Eager and Lazy Aggregation transformations can be extended to the case of SJ and SGB as shown in the Theorem 3 (Section 4.4.3).

Pushing Similarity Predicate from Join-Around to Group-by:

R120. The similarity predicate of the Join-Around can be completely pushed down to a grouping operator as specified in Section 4.4.4.

Pushing Similarity Predicate from Ɛ-Join to Group-by:

R121. The similarity predicate of the Ɛ-Join can be partially pushed down to a grouping operator as specified in Section 4.4.5.

### Distribution of Selection and Similarity Selection over SGB (SGB-U, SGB-A, SGB-D)

R122. $\sigma_{\theta(G)}\left({}_{(G,S)}\Gamma_{F(A)}(E)\right) \not\equiv {}_{(G,S)}\Gamma_{F(A)}(\sigma_{\theta(G)}(E))$.
R123. $\sigma_{\theta_{\varepsilon,C}(G)}\left({}_{(G,S)}\Gamma_{F(A)}(E)\right) \not\equiv {}_{(G,S)}\Gamma_{F(A)}(\sigma_{\theta_{\varepsilon,C}(G)}(E))$.
R124. $\sigma_{\theta_{kNN,C}(G)}\left({}_{(G,S)}\Gamma_{F(A)}(E)\right) \not\equiv {}_{(G,S)}\Gamma_{F(A)}(\sigma_{\theta_{kNN,C}(G)}(E))$.

### Distribution of Similarity Selection over ∪, ∩ and –

R125. $\sigma_{\theta_{\varepsilon,C}(e)}(E_1 \cup E_2) \equiv (\sigma_{\theta_{\varepsilon,C}(e)}(E_1)) \cup (\sigma_{\theta_{\varepsilon,C}(e)}(E_2))$.

R126. $\sigma_{\theta_{\varepsilon,C(e)}}(E_1 \cap E_2) \equiv (\sigma_{\theta_{\varepsilon,C(e)}}(E_1)) \cap (\sigma_{\theta_{\varepsilon,C(e)}}(E_2))$.
R127. $\sigma_{\theta_{\varepsilon,C(e)}}(E_1 - E_2) \equiv (\sigma_{\theta_{\varepsilon,C(e)}}(E_1)) - (\sigma_{\theta_{\varepsilon,C(e)}}(E_2))$.
R128. $\sigma_{\theta_{kNN,C(e)}}(E_1 \cup E_2) \not\equiv (\sigma_{\theta_{kNN,C(e)}}(E_1)) \cup (\sigma_{\theta_{kNN,C(e)}}(E_2))$.
R129. $\sigma_{\theta_{kNN,C(e)}}(E_1 \cap E_2) \not\equiv (\sigma_{\theta_{kNN,C(e)}}(E_1)) \cap (\sigma_{\theta_{kNN,C(e)}}(E_2))$.
R130. $\sigma_{\theta_{kNN,C(e)}}(E_1 - E_2) \not\equiv (\sigma_{\theta_{kNN,C(e)}}(E_1)) - (\sigma_{\theta_{kNN,C(e)}}(E_2))$.
R131. $\sigma_{\theta_{\varepsilon,C(e)}}(E_1 \cup E_2) \not\equiv (\sigma_{\theta_{\varepsilon,C(e)}}(E_1)) \cup E_2$.
R132. $\sigma_{\theta_{\varepsilon,C(e)}}(E_1 \cap E_2) \equiv (\sigma_{\theta_{\varepsilon,C(e)}}(E_1)) \cap E_2$.
R133. $\sigma_{\theta_{\varepsilon,C(e)}}(E_1 - E_2) \equiv (\sigma_{\theta_{\varepsilon,C(e)}}(E_1)) - E_2$.
R134. $\sigma_{\theta_{kNN,C(e)}}(E_1 \cup E_2) \not\equiv (\sigma_{\theta_{kNN,C(e)}}(E_1)) \cup E_2$.
R135. $\sigma_{\theta_{kNN,C(e)}}(E_1 \cap E_2) \not\equiv (\sigma_{\theta_{kNN,C(e)}}(E_1)) \cap E_2$.
R136. $\sigma_{\theta_{kNN,C(e)}}(E_1 - E_2) \not\equiv (\sigma_{\theta_{kNN,C(e)}}(E_1)) - E_2$.

### Distribution of Projection over Similarity Join

If $\theta_S$ involves only attributes of $L_1 \cup L_2$, and additionally for k-based operations, $E.PrimKey \in L_1$ and $F.PrimKey \in L_2$:

R137. $\Pi_{L_1 \cup L_2}\big(E \bowtie_{\theta_{\varepsilon}(e,f)} F\big) \equiv (\Pi_{L_1}(E)) \bowtie_{\theta_{\varepsilon}(e,f)} (\Pi_{L_2}(F))$.
R138. $\Pi_{L_1 \cup L_2}\big(E \bowtie_{\theta_{kNN}(e,f)} F\big) \equiv (\Pi_{L_1}(E)) \bowtie_{\theta_{kNN}(e,f)} (\Pi_{L_2}(F))$.
R139. $\Pi_{L_1 \cup L_2}\big(E \bowtie_{\theta_{kD}(e,f)} F\big) \equiv (\Pi_{L_1}(E)) \bowtie_{\theta_{kD}(e,f)} (\Pi_{L_2}(F))$.
R140. $\Pi_{L_1 \cup L_2}\big(E \bowtie_{\theta_{A}(e,f)} F\big) \equiv (\Pi_{L_1}(E)) \bowtie_{\theta_{A}(e,f)} (\Pi_{L_2}(F))$.

If $L_1$ and $L_2$ are sets of attributes from $E$ and $E$, respectively; $L_3$ contains attributes that are involved in the join predicate but are not in $L_1 \cup L_2$; $L_4$ contains attributes that are involved in the join predicate but are not in $L_1 \cup L_2$; and additionally for k-based operations, $E.Primkey \in (L_1 \cup L_3)$; and $F.Primkey \in (L_2 \cup L_4)$:

R141. $\Pi_{L_1 \cup L_2}\big(E \bowtie_{\theta_{\varepsilon}(e,f)} F\big) \equiv \Pi_{L_1 \cup L_2}((\Pi_{L_1 \cup L_3}(E)) \bowtie_{\theta_{\varepsilon}(e,f)} (\Pi_{L_2 \cup L_4}(F)))$.
R142. $\Pi_{L_1 \cup L_2}\big(E \bowtie_{\theta_{kNN}(e,f)} F\big) \equiv \Pi_{L_1 \cup L_2}((\Pi_{L_1 \cup L_3}(E)) \bowtie_{\theta_{kNN}(e,f)} (\Pi_{L_2 \cup L_4}(F)))$.
R143. $\Pi_{L_1 \cup L_2}\big(E \bowtie_{\theta_{kD}(e,f)} F\big) \equiv \Pi_{L_1 \cup L_2}((\Pi_{L_1 \cup L_3}(E)) \bowtie_{\theta_{kD}(e,f)} (\Pi_{L_2 \cup L_4}(F)))$.
R144. $\Pi_{L_1 \cup L_2}\big(E \bowtie_{\theta_{A}(e,f)} F\big) \equiv \Pi_{L_1 \cup L_2}((\Pi_{L_1 \cup L_3}(E)) \bowtie_{\theta_{A}(e,f)} (\Pi_{L_2 \cup L_4}(F)))$.

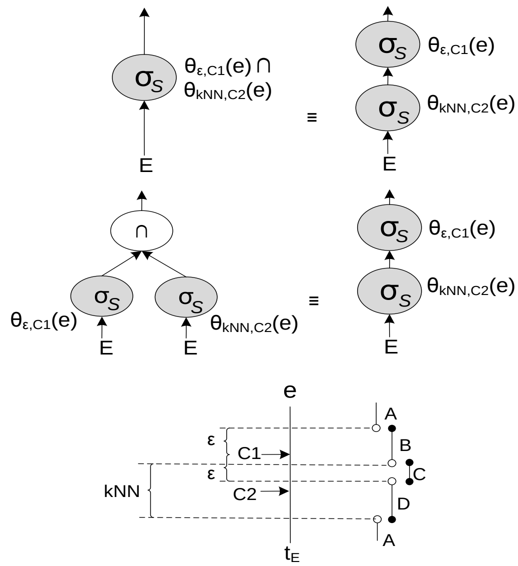


Fig. 44. Combining Ɛ-Selection and kNN-Selection (R2) – Proof

## B. ADDITIONAL PROOFS

PROOF SKETCH OF RULE R2. Consider a generic tuple $t_E$ of $E$. We will show that for any possible value of $t_E$, the results generated by the plans of both sides of the rule are the same. The top part of Fig. 44 gives a graphical representation of Rule R2. Using the conceptual evaluation order of similarity queries, we can transform the left part of the rule to an equivalent expression that uses the intersection operation as represented in the middle part of Fig. 44. We will use this second version of the rule in the remaining part of the proof. The bottom part of Fig. 44 gives the different possible regions for the value of $t_E.e$. Note that the region marked as kNN, which comprises regions C and D, represents the region that contains the kNN closest neighbors of *C2*.

1. When the value of $t_E.e$ belongs to A. In the LHS plan, $t_E$ is not selected in any of the selection operators since it does not satisfy any of the Similarity Selection predicates. Thus, no output is generated by this plan. In the RHS plan, $t_E$ is filtered out by the kNN-Selection. No tuple flows to the Ɛ-Selection. Thus, no output is generated by this plan either.
2. When the value of $t_E.e$ belongs to B. In the LHS plan, $t_E$ is selected in the Ɛ-Selection but not in the kNN-Selection. The intersection operator does not produce any output and consequently no output is generated by this plan. In the RHS plan, $t_E$ is filtered out by the kNN-Selection. No tuple flows to the Ɛ-Selection. Thus, no output is generated by this plan either.
3. When the value of $t_E.e$ belongs to C. In the LHS plan, $t_E$ is selected by both Similarity Selection operators. Consequently, $t_E$ belongs to the output of the intersection operator. $t_E$ belongs to the output of the LHS plan. In the RHS plan, $t_E$ is selected by the kNN-Selection. $t_E$ is also selected by the Ɛ-Selection. Thus, $t_E$ also belongs to the output of the RHS plan.
4. When the value of $t_E.e$ belongs to D. In the LHS plan, $t_E$ is selected in the kNN-Selection but not in the Ɛ-Selection. The intersection operator does not produce any output and consequently no output is generated by this plan. In the RHS plan, $t_E$ is selected by the kNN-Selection but filtered out by the Ɛ-Selection. Thus, no output is generated by this plan either.

□

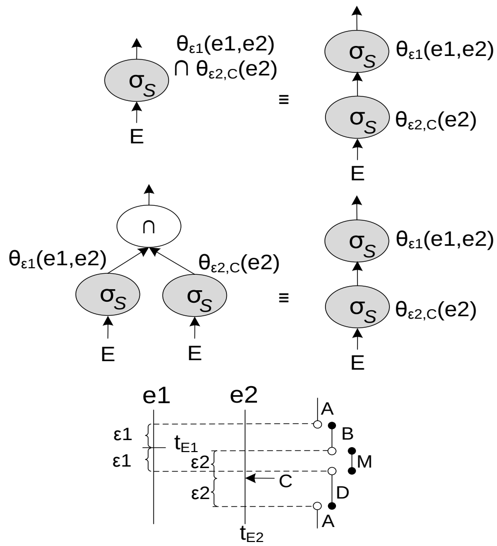


Fig. 45. Combining Ɛ-Join and Ɛ-Selection (R5) – Proof

PROOF SKETCH OF RULE R5. Assume that $\theta_{\varepsilon 1}(e1,e2)$ is defined over relations $E_1$ and $E_2$, and that the input relation $E$ is the cross product of all the relations involved in the similarity-aware predicates, i.e., $E = E_1 \times E_2$. Furthermore, assume that the join attributes are $E_1.e_1$ and $E_2.e_2$. Consider a generic tuple $t_{E1}$ of $E_1$. We will show that for any possible pair $(t_{E1},t_{E2})$, where $t_{E2}$ is a tuple of $E_2$, the results generated by the plans of both sides of the rule are the same (we consider the first equivalence of R5). The top part of Fig. 45 gives a graphical representation of Rule R5. Using the conceptual evaluation order of similarity queries, we can transform the left part of the rule to an equivalent expression that uses the intersection operation as represented in the middle part of Fig. 45. We will use this second version of the rule in the remaining part of the proof. The bottom part of Fig. 45 gives the different possible regions for the value of $t_{E2}.e_2$.

1. When the value of $t_{E2}.e_2$ belongs to A. In the LHS plan, the pair $(t_{E1},t_{E2})$ is not selected in any similarity-aware operator since it does not satisfy any of their predicates. Thus, no output is generated by this plan. In the RHS plan, $(t_{E1},t_{E2})$ is filtered out by the bottom selection since $dist(t_{E2}.e_2, C) > \mathcal{E}2$. No tuple flows to the top operator. Thus, no output is generated by this plan either.
2. When the value of $t_{E2}.e_2$ belongs to B. In the LHS plan, the pair $(t_{E1},t_{E2})$ is selected in the left Similarity Selection but not in the right one. The intersection operator does not produce any output and consequently no output is generated by this plan. In the RHS plan, $(t_{E1},t_{E2})$ is filtered out by the bottom selection since $dist(t_{E2}.e_2, C) > \mathcal{E}2$. No tuple flows to the top operator. Thus, no output is generated by this plan either.
3. When the value of $t_{E2}.e_2$ belongs to M. In the LHS plan, the pair $(t_{E1},t_{E2})$ is selected in both similarity-aware operators. Consequently, $(t_{E1},t_{E2})$ belongs to the output of the intersection operator. $(t_{E1},t_{E2})$ belongs to the output of the LHS plan. In the RHS plan, $(t_{E1},t_{E2})$ is selected by the bottom selection since $dist(t_{E2}.e_2,C) \leq \mathcal{E}2$. $(t_{E1},t_{E2})$ is also selected by the top selection since $dist(t_{E1}.e_1,t_{E2}.e_2) \leq \mathcal{E}1$. Thus, the pair $(t_{E1},t_{E2})$ belongs also to the output of the RHS plan.

   When the value of $t_{E2}.e_2$ belongs to D. In the LHS plan, the pair $(t_{E1},t_{E2})$ is selected in the right Similarity Selection but not in the left one. The intersection operator does not produce any output and consequently no output is generated by this plan. In the RHS plan, $(t_{E1},t_{E2})$ is selected in the bottom selection since $dist(t_{E2}.e_2, C) \leq \mathcal{E}2$ but it is filtered out by the top selection. Thus, no output is generated by this plan either. ☐

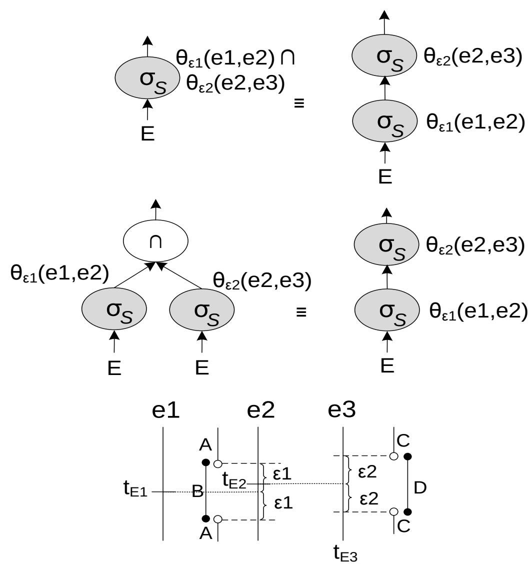


Fig. 46. Combining/separating two Ɛ-Join predicates (R32) – Proof

PROOF SKETCH OF RULE R32. Assume that $\theta_{\varepsilon 1}(e1,e2)$ is defined over relations $E_1$ and $E_2$, and $\theta_{\varepsilon 2}(e2,e3)$ over relations $E_2$ and $E_3$. Assume also that the input relation $E$ is the cross product of all the relations involved in the similarity-aware predicates, i.e., $E = E_1 \times E_2 \times E_3$. Furthermore, assume that the join attributes in $\theta_{\varepsilon 1}$ are $E_1.e_1$ and $E_2.e_2$, and in $\theta_{\varepsilon 2}$ are $E_2.e_2$ and $E_3.e_3$. Consider a generic tuple $t_{E1}$ of $E_1$. We will show that for any possible triplet $(t_{E1},t_{E2},t_{E3})$, where $t_{E2}$ is a tuple of $E_2$, and $t_{E3}$ is a tuple of $E_3$, the results generated by the plans of both sides of the rule are the same (we consider the equivalence between the first and third components of R32). The top part of Fig. 46 gives a graphical representation of Rule R32. Using the conceptual evaluation order of similarity queries, we can transform the left part of the rule to an equivalent expression that uses the intersection operation as represented in the middle part of Fig. 46. We will use this second version of the rule in the remaining part of the proof. The bottom part of Fig. 46 gives the different possible regions for the values of $t_{E2}.e_2$ and $t_{E3}.e_3$. Note that the regions for $t_{E3}.e_3$ have been specified based on a generic tuple $t_{E2}$ with $t_{E2}.e_2$ in region B.

1. When the value of $t_{E2}.e_2$ belongs to A. In the LHS plan, the triplet $(t_{E1},t_{E2},t_{E3})$ is not selected in any similarity-aware operator since it does not satisfy any of their predicates. Thus, no output is generated by this plan. In the RHS plan, $(t_{E1},t_{E2},t_{E3})$ is filtered out by the bottom selection since $dist(t_{E1}.e_1,t_{E2}.e_2)>\mathcal{E}1$. No tuple flows to the top operator. Thus, no output is generated by this plan either.
2. When the value of $t_{E2}.e_2$ belongs to B and the value of $t_{E3}.e_3$ belongs to C. In the LHS plan, the triplet $(t_{E1},t_{E2},t_{E3})$ is selected in the left Similarity Selection since $dist(t_{E1}.e_1,t_{E2}.e_2)\leq\mathcal{E}1$ but not in the right one since $dist(t_{E2}.e_2,t_{E3}.e_3)>\mathcal{E}2$. The intersection operator does not produce any output and consequently no output is generated by this plan. In the RHS plan, $(t_{E1},t_{E2},t_{E3})$ is selected in the bottom selection since $dist(t_{E1}.e_1,t_{E2}.e_2)\leq\mathcal{E}1$ but it is filtered out by the top selection since $dist(t_{E2}.e_2,t_{E3}.e_3)>\mathcal{E}2$. Thus, no output is generated by this plan either.
3. When the value of $t_{E2}.e_2$ belongs to B and the value of $t_{E3}.e_3$ belongs to D. In the LHS plan, the triplet $(t_{E1},t_{E2},t_{E3})$ is selected in both similarity-aware operators since $dist(t_{E1}.e_1,t_{E2}.e_2)\leq\mathcal{E}1$ (left) and $dist(t_{E2}.e_2,t_{E3}.e_3)\leq\mathcal{E}2$ (right). Consequently, $(t_{E1},t_{E2},t_{E3})$ belongs to the output of the intersection operator. $(t_{E1},t_{E2},t_{E3})$ belongs to the output of the LHS plan. In the RHS plan, $(t_{E1},t_{E2},t_{E3})$ is selected by the bottom selection since $dist(t_{E1}.e_1,t_{E2}.e_2)\leq\mathcal{E}1$. $(t_{E1},t_{E2},t_{E3})$ is also selected by the top selection since $dist(t_{E2}.e_2,t_{E3}.e_3)\leq\mathcal{E}2$. Thus, $(t_{E1},t_{E2},t_{E3})$ also belongs to the output of the RHS plan. □

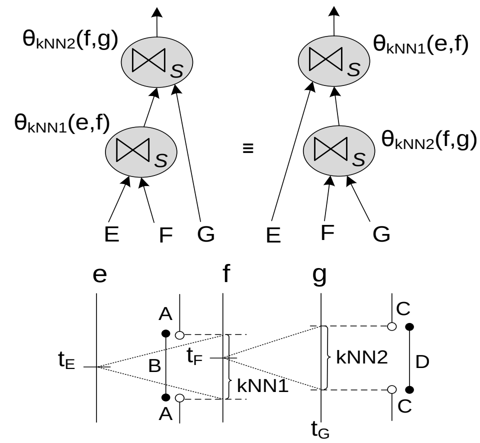


Fig. 47. Associativity of kNN-Join operators - when attributes in predicates have a single direction: e1→e2, e2→e3 (R103) –Proof

PROOF SKETCH OF RULE R66. In the LHS expression of the equivalence, all the join links satisfy $dist(e_1,e_2) \leq \varepsilon$. Given that the distance function *dist* is symmetric, $dist(e_1,e_2)=dist(e_2,e_1)$. Thus, the condition $dist(e_2,e_1) \leq \varepsilon$ in the LHS expression will produce the same set of join links. □

PROOF SKETCH OF RULE R103. Assume that the join attributes in $\theta_{kNN1}$ are *E.e* and *F.f* and the join attributes in $\theta_{kNN2}$ are *F.f* and *G.g*. Consider a generic tuple $t_E$ of *E*. We will show that for any possible triplet $(t_E,t_F,t_G)$, where $t_F$ is a tuple of *F* and $t_G$ is a tuple of *G*, the results generated by the plans of both sides of the rule are the same. The top part of Fig. 47 gives a graphical representation of Rule R103. The bottom part of this figure gives the different possible regions for the values of $t_F.f$ and $t_G.g$. Note that the regions for $t_G.g$ have been specified based on a generic tuple $t_F$ with $t_F.f$ in region B. The region marked as *kNN1* represents the segment that contains the *kNN1* closest neighbors of $t_E$ in *F*. The region marked as *kNN2* represents the segment that contains the *kNN2* closest neighbors of $t_F$ in *G*. Note that for a given kNN-Join ($\theta_{kNN1}$ or $\theta_{kNN2}$) and a given outer tuple *t*, the join identifies the same set of *k* nearest neighbors of *t* in both plans. This is the case since (i) kNN-Join over $R_1$ and $R_2$ makes use of primary keys in both input relations ($R_1.pk_1$, $R_2.pk_2$) and ignores tuples in $R_2$ that have the same primary key, and (ii) the set of different values of $R_2.pk_2$ in the inner input of both plans is the same. Furthermore, note that the set of different values of $R_2.pk_2$ in the inner input of both plans corresponds to the set of all different values of $R_2.pk_2$ in the base relation $R_2$.

1. When the value of $t_F.f$ belongs to B and the value of $t_G.g$ belongs to D. In the LHS plan, the pair $(t_E,t_F)$ belongs to the output of the bottom kNN-Join ($\theta_{kNN1}$) since $t_F$ is one of the *kNN1* closest neighbors of $t_E$ in *F*. $(t_E,t_F)$ flows to the top kNN-Join. The triplet $(t_E,t_F,t_G)$ belongs also to the output of the top kNN-Join ($\theta_{kNN2}$) since $t_G$ is one of the *kNN2* closest neighbors of $t_F$ in *G*. Consequently, $(t_E,t_F,t_G)$ belongs to the output of the LHS plan. In the RHS plan, $(t_F,t_G)$ belongs to the output of the bottom kNN-Join ($\theta_{kNN2}$) since $t_G$ is one of the *kNN2* closest neighbors of $t_F$ in *G*. The triplet $(t_E,t_F,t_G)$ belongs also to the output of the top kNN-Join ($\theta_{kNN1}$) since $t_F$ is one of the *kNN1* closest neighbors of $t_E$ in *F*. Thus, $(t_E,t_F,t_G)$ belongs also to the output of the RHS plan. Note that in the RHS plan, the bottom join ($\theta_{kNN2}$) matches each inner tuple of *F* to its closest *kNN2* neighbors in *G*. The output of this join will contain all the values of $F.pk_2$ (the primary key of *F*) in the base relation *F*. Consequently, the set of all different values of $F.pk_2$ in the inner input of $\theta_{kNN1}$ is the same in both plans. Therefore, for a given inner tuple *t*, the join $\theta_{kNN1}$ will find the same set of *kNN1* nearest neighbors of *t* in both plans.
2. When the value of $t_F.f$ belongs to B and the value of $t_G.g$ belongs to C. In the LHS plan, the pair $(t_E,t_F)$ belongs to the output of the bottom kNN-Join ($\theta_{kNN1}$) since $t_F$ is one of the *kNN1*

closest neighbors of $t_E$ in $F$. ($t_E$,$t_F$) flows to the top kNN-Join. However, the triplet ($t_E$,$t_F$,$t_G$) does not belong to the output of the top kNN-Join ($\theta_{kNN2}$) since $t_G$ is not one of the *kNN2* closest neighbors of $t_F$ in $G$. Consequently, no output is generated by this plan. In the RHS plan, ($t_F$,$t_G$) does not belongs to the output of the bottom kNN-Join ($\theta_{kNN2}$) since $t_G$ is not one of the *kNN2* closest neighbors of $t_F$ in $G$. No tuple flows to the top join. Thus, no output is generated by this plan either.

3. When the value of $t_F.f$ belongs to A. In the LHS plan, the pair ($t_E$,$t_F$) does not belongs to the output of the bottom kNN-Join ($\theta_{kNN1}$) since $t_F$ is not one of the *kNN1* closest neighbors of $t_E$ in $F$. No tuple flows to the top join. Consequently no output is generated by this plan. In the RHS plan, ($t_F$,$t_G$) may or may not belong to the output of the bottom kNN-Join ($\theta_{kNN2}$). However, any triplet ($t_E$,$t_F$,$t_G$) does not belong to the output of the top kNN-Join ($\theta_{kNN1}$) since $t_F$ is not one of the *kNN1* closest neighbors of $t_E$ in $F$. Thus, no output is generated by this plan either. □

PROOF SKETCH OF RULE R115 (GR15). Note that pushing Ɛ-Selection under the outer input of the Ɛ-Join has been already studied in Rule R86. We focus here on the validity of pushing the Ɛ-Selection operation under the inner input of the Ɛ-Join. Assume that the selection predicate $\theta_{\varepsilon 1,C}(e)$ is $dist(e,C) \leq \varepsilon 1$ and the join predicate $\theta_{\varepsilon 2}(e,f)$ is $dist(e,f) \leq \varepsilon 2$.

1. Due to Triangular Inequality, $dist(f,C) \leq dist(f,e) + dist(e,C)$.
2. Due to Commutativity, we have that $dist(f,C) \leq dist(e,f) + dist(e,C)$.
3. Using in (2) the fact that $dist(e,f) \leq \varepsilon 2$, $dist(f,C) \leq \varepsilon 2 + dist(e,C)$.
4. Using in (3) the fact that $dist(e,C) \leq \varepsilon 1$, $dist(f,C) \leq \varepsilon 1 + \varepsilon 2$.

The expression in (4) $dist(f,C) \leq \varepsilon 1 + \varepsilon 2$ represents an Ɛ-Selection predicate that can be applied on $f$. This predicate is in fact the predicate being applied on $f$ in the inner input of the RHS part of Rule R115. □

PROOF SKETCH OF RULE R116 (GR16). Assume that in the LHS part of Rule R116, the join predicate $\theta_{\varepsilon 1}(e,f)$ is $dist(e,f) \leq \varepsilon 1$, and the join predicate $\theta_{\varepsilon 2}(f,g)$ is $dist(f,g) \leq \varepsilon 2$. The order of attributes in these expressions is irrelevant because the distance function is Commutative.

1. Due to Triangular Inequality, $dist(e,g) \leq dist(e,f)+dist(f,g)$.
2. Since $dist(e,f) \leq \varepsilon 1$ and $dist(f,g) \leq \varepsilon 2$, $dist(e,g) \leq \varepsilon 1+\varepsilon 2$.

The expression in (2) $dist(e,g) \leq \varepsilon 1+\varepsilon 2$ represents a join predicate that can be applied on $e$ and $g$. This predicate is in fact the predicate being applied on $e$ and $g$ in the left join of the RHS part of Rule R116. Note that the RHS part of the rule requires a second join that applies the two join predicates of the LHS part because some tuples that do not satisfy these predicates can be present in the output of the join between $e$ and $g$. □

PROOF SKETCH OF THEOREM 1. Consider a group $G_d$ generated by $g\ [NGA_d,\ Seg_d]\sigma[C_d]r_d$ for some instance $r_d$ of $R_d$. Due to conditions (iii) and (iv), all the rows of $G_d$ have the same values of $GA_d$ and the joining attributes. Every tuple of $G_d$ joins with the same set of tuples $SA_u(G_d)$. Let $S_u(G_d)$ be the subset of $SA_u(G_d)$ that has a unique value of $GA_u$. Consider two groups of $g\ [NGA_d,\ Seg_d]\sigma[C_d]r_d$: $R_{d1}$ and $R_{d2}$. There are two cases to be considered.

*Case 1*: $G_{d1}[GA_d] \sim G_{d2}[GA_d]$ *and* $S_u(G_{d1})[GA_u] \sim S_u(G_{d2})[GA_u]$. In $E_2$, the results of the join operations represented by the following two expressions are merged into the same similarity group by the second SGB.

1. $((F_{d1}[AA_d],\ COUNT)\pi[NGA_d,\ GA_d^+,\ AA_d]G_{d1}) \times S_u(G_{d1})$.
2. $((F_{d1}[AA_d],\ COUNT)\pi[NGA_d,\ GA_d^+,\ AA_d]G_{d2}) \times S_u(G_{d2})$.

In $E_1$, each row of $G_{d1}$ and $G_{d2}$ joins with $S_u(G_{d1})$ and $S_u(G_{d2})$ respectively and all the resulting rows are also merged by the second SGB. Due to condition (i), the aggregation values in the resulting row of the following expressions in $E_1$ and $E_2$ respectively are the same.

3. $F_d[AA_d]\pi_A[GA_d,GA_u,AA_d]\ ((G_{d1} \times S_u(G_{d1})) \cup_A (G_{d2} \times S_u(G_{d2})))$.
4. $F_{d2}[FAA_d]\pi_A[GA_d,GA_u,FAA_d](((F_{d1}[AA_d]\pi_A[NGA_d,\ GA_d^+,\ AA_d]G_{d1}) \times S_u(G_{d1}))$
   $\cup_A ((F_{d1}[AA_d]\pi_A[NGA_d,\ GA_d^+,\ AA_d]G_{d2}) \times S_u(G_{d2}))$.

Due to condition (ii), the aggregation values in the resulting row of the following expressions in $E_1$ and $E_2$, respectively, are the same.

5. $F_u[AA_u]\pi_A[GA_d,GA_u,AA_u]\ ((G_{d1} \times S_u(G_{d1})) \cup_A (G_{d2} \times S_u(G_{d2})))$.
6. $F_{ua}[AA_u,CNT]\pi_A[GA_d,GA_u,\ AA_u,\ CNT](((COUNT\ \pi_A[NGA_d,\ GA_d^+]G_{d1}) \times S_u(G_{d1}))$
   $\cup_A ((COUNT\ \pi_A[NGA_d,\ GA_d^+]G_{d2}) \times S_u(G_{d2}))$.

*Case 2:* $G_{d1}[GA_d]\ !\sim\ G_{d2}[GA_d]$ *or* $S_u(G_{d1})[GA_u]\ !\sim\ S_u(G_{d2})[GA_u]$. In $E_2$, the results of the join operations represented by (1) and (2) are not merged into the same similarity group by the second SGB. In $E_1$, each row of $G_{d1}$ and $G_{d2}$ joins with $S_u(G_{d1})$ and $S_u(G_{d2})$, respectively, but the resulting rows are not merged by the second SGB. Due to condition (i), the aggregation values in the resulting row of the following expressions in $E_1$ and $E_2$, respectively, are the same.

7. $F_d[AA_d]\pi_A[GA_d,GA_u,AA_d](G_{d1} \times S_u(G_{d1}))$.
8. $F_{d2}[FAA_d]\pi_A[GA_d,GA_u,FAA_d]((F_{d1}[AA_d]\pi_A[NGA_d,\ GA_d^+,\ AA_d]G_{d1}) \times S_u(G_{d1}))$.

Due to condition (ii), the aggregation values in the resulting row of the following expressions in $E_1$ and $E_2$, respectively, are the same.

9. $F_u[AA_u]\pi_A[GA_d,GA_u,AA_u]\ ((G_{d1} \times S_u(G_{d1}))$.
10. $F_{ua}[AA_u,CNT]\pi_A[GA_d,GA_u,\ AA_u,\ CNT]((COUNT\ \pi_A[NGA_d,\ GA_d^+]G_{d1}) \times S_u(G_{d1}))$. □

PROOF SKETCH OF THEOREM 2. The validity of this theorem relies on the following properties.

P1. Given $R_d'$ and $R_u'$ instances of $R_d$ and $R_u$ respectively, the result of $(R_d' \mathbin{\tilde{\bowtie}}_{C0} R_u')$ is equivalent to the result of $(R_d' \bowtie_\theta R_u')$ where $\theta$ = disjunction of $(R_d.C0_d{=}x \wedge R_u.C0_u{=}y)$ for every different link $(x,y)$ of the result of $(R_d' \mathbin{\tilde{\bowtie}}_{C0} R_u')$.

P2. $\theta$, as defined in P1, remains unchanged and valid when $R_d'$ is augmented with tuples that have already present values of $R_d'.C0_d$, i.e., duplicates, or when such tuples are removed from $R_d'$.

The validity of this theorem can be shown by following these steps:

For every $R_{d'}$ and $R_{u'}$ instances of $R_d$ and $R_u$, respectively,

1. $E_1$: $F[AA_d,\ AA_u]\pi_A[GA_d,\ GA_u,\ AA_d,\ AA_u]\ g\ [GA_d,\ GA_u]\sigma[C_d \wedge C_u]\ (R_{d'} \mathbin{\tilde{\bowtie}}_{C0} R_{u'})$,
   is equivalent to:

   $E_{1'}$: $F[AA_d,\ AA_u]\pi_A[GA_d,\ GA_u,\ AA_d,\ AA_u]\ g\ [GA_d,\ GA_u]\sigma[C_d \wedge C_u]\ (R_{d'} \bowtie_\theta R_{u'})$,
   where $\theta$ is defined as in P1.
2. $E_{1'}$: $F[AA_d,\ AA_u]\pi_A[GA_d,\ GA_u,\ AA_d,\ AA_u]\ g\ [GA_d,\ GA_u]\sigma[C_d \wedge C_u]\ (R_{d'} \bowtie_\theta R_{u'})$,
   is equivalent to:

Table VI. Common distance functions

| Distance Function | Definition |
|---|---|
| p-norm Distance | p-norm distance of two vectors $(x_1, x_2, ..., x_n)$ and $(y_1, y_2, ..., y_n)$ is defined as: 1-norm distance = $\sum_{i=1}^{n} \lvert x_i - y_i \rvert$ 2-norm distance = $\left(\sum_{i=1}^{n} \lvert x_i - y_i \rvert^2\right)^{1/2}$ p-norm distance = $\left(\sum_{i=1}^{n} \lvert x_i - y_i \rvert^p\right)^{1/p}$ infinity-norm distance = $\lim_{p\to\infty}\left(\sum_{i=1}^{n} \lvert x_i - y_i \rvert^p\right)^{1/p}$ |
| Cosine Distance 1 | CD1(A,B) = 1 - CS(A,B), where A and B are vectors and CS(A,B) is the Cosine Similarity. CS(A,B) = (A ·B)/(‖A‖‖B‖) |
| Cosine Distance 2 | CD2(A,B) = arccos(CS(A,B)) |
| Discrete Metric Function | DM(x,y) = 0 if x = y, 1 otherwise, where x and y are numbers. |
| Longest Common Subsequence | LCS(X,Y) = longest subsequence common to strings or time series X and Y. |
| Edit Distance with Equal Weights | ED(X,Y) = minimum number of operations needed to transform string X into string Y. Allowed operations: insertion, deletion, and substitution of a single character. |
| Edit Distance with Different Weights | ED(X,Y) = min(w(E)), where E is a sequence of edit operations that transforms string X into string Y, and w is a weight function that assigns a nonnegative real number w(x, y) to each elementary edit operation. |
| Hamming Distance | HD(X,Y) = number of positions in which the characters of strings X and Y are different. |
| Jaccard Distance | JD(A,B) = 1- JS(A,B), where JS(A,B) = (∣A∩B∣/∣A∪B∣). A and B are two generic sets. For string data, JS(A,B) = number of shared tokens/total number of tokens. For vector data, JS(A,B)=number of matching cells/total number of cells. |

$E_{2'}$: $\pi_D[GA_d, GA_u, FAA](F_{ua}[AA_u,CNT], F_{d2}[FAA_d])$
$\pi_A[GA_d, GA_u, AA_u, FAA_d, CNT]\ g\ [GA_d, GA_u]\sigma[C_u]$
$(((F_{d1}[AA_d], COUNT)\pi_A[NGA_d, GA_d^+, AA_d]\ g\ [NGA_d]\sigma[C_d]R_{d'}) \bowtie_\theta R_{u'})$,

because of the eager and lazy aggregation main theorem for regular operators.

3. $E_{2'}$: $\pi_D[GA_d, GA_u, FAA](F_{ua}[AA_u,CNT], F_{d2}[FAA_d])$
$\pi_A[GA_d, GA_u, AA_u, FAA_d, CNT]\ g\ [GA_d, GA_u]\sigma[C_u]$
$(((F_{d1}[AA_d], COUNT)\pi_A[NGA_d, GA_d^+, AA_d]\ g\ [NGA_d]\sigma[C_d]R_{d'}) \bowtie_\theta R_{u'})$,

is equivalent to:

$E_2$: $\pi_D[GA_d, GA_u, FAA](F_{ua}[AA_u,CNT], F_{d2}[FAA_d])$
$\pi_A[GA_d, GA_u, AA_u, FAA_d, CNT]\ g\ [GA_d, GA_u]\sigma[C_u]$
$(((F_{d1}[AA_d], COUNT)\pi_A[NGA_d, GA_d^+, AA_d]\ g\ [NGA_d]\sigma[C_d]R_{d'}) \widetilde{\bowtie}_{C0} R_{u'})$,

since the grouping operation before the join merges only tuples that share the same value of $R_{d'}.C0_d$, and P2. □

PROOF SKETCH OF THEOREM 3. The validity of this theorem relies on the validity of Theorem 1 and Theorem 2. □

### C. DEFINITION OF COMMON DISTANCE FUNCTIONS

Table VI shows the definition of common distance functions.