A Formal Counterexample to the Cost-Preserving Single-Source Unsplittable Flow Conjecture

Arthur Freitas Ramos David Barros Hulak
Ruy J. G. B. de Queiroz

August 7, 2026

Abstract

Dinitz, Garg, and Goemans proved that a feasible fractional single-source flow can be rounded to an unsplittable flow with additive congestion bounded by the largest demand [1]. Goemans conjectured that the rounding can simultaneously preserve cost; recent work still treats this as a central open cost variant [3].

This entry verifies the finite counterexample announced by Dmitry Rybin in July 2026 [2]. It defines a generic finite path-flow model and realizes the displayed instance as a directed graph with seven vertices, nine arcs, and three terminal demands. Isabelle classifies every source-to-terminal path and checks a feasible fractional flow of cost 58 that saturates every arc. The maximum demand is 15. Any unsplittable routing whose load on every arc is at most the fractional load plus 15 must choose at least two positive-cost routes, each costing 30, and therefore costs at least 60. Consequently no cost-preserving rounding exists, even under the weaker non-strict congestion bound.

AI assistance was used for proof engineering. The final definitions, statements, and proofs are checked by Isabelle.

Contents

theory *Dinitz-Garg-Goemans-Counterexample*
  imports *Complex-Main*
begin

1 Finite path-flow instances

We use a path-based formulation of single-source flow. A route records one admissible source-to-terminal path, and *pf-uses* is its arc-incidence predicate. The finite carrier types make all loads and costs explicit finite sums.

record ($'commodity$, $'route$, $'arc$) *path-flow-instance* =
pf-admissible :: $'commodity \Rightarrow 'route \Rightarrow bool$
pf-demand :: $'commodity \Rightarrow real$
pf-capacity :: $'arc \Rightarrow real$
pf-cost :: $'arc \Rightarrow real$
pf-uses :: $'commodity \Rightarrow 'route \Rightarrow 'arc \Rightarrow bool$

definition *fractional-load* ::
($'commodity::finite$, $'route::finite$, $'arc$) *path-flow-instance* $\Rightarrow$
($'commodity \Rightarrow 'route \Rightarrow real) \Rightarrow 'arc \Rightarrow real$ **where**
fractional-load I f a =
$(\sum k\in UNIV.\ \sum r\in UNIV.\ \textit{if pf-uses I k r a then f k r else 0})$

definition *fractional-cost* ::
($'commodity::finite$, $'route::finite$, $'arc::finite$) *path-flow-instance* $\Rightarrow$
($'commodity \Rightarrow 'route \Rightarrow real) \Rightarrow real$ **where**
fractional-cost I f =
$(\sum a\in UNIV.\ \textit{pf-cost I a} * \textit{fractional-load I f a})$

definition *maximum-demand* ::
($'commodity::finite$, $'route$, $'arc$) *path-flow-instance* $\Rightarrow real$ **where**
maximum-demand I = *Max* (*pf-demand I* ‘ ($UNIV$ :: $'commodity\ set$))

definition *well-formed-instance* ::
($'commodity$, $'route$, $'arc$) *path-flow-instance* $\Rightarrow bool$ **where**
well-formed-instance I $\longleftrightarrow$
$(\forall k.\ 0 < \textit{pf-demand I k} \wedge (\exists r.\ \textit{pf-admissible I k r})) \wedge$
$(\forall a.\ 0 \le \textit{pf-capacity I a} \wedge 0 \le \textit{pf-cost I a})$

definition *fractional-feasible* ::
($'commodity::finite$, $'route::finite$, $'arc$) *path-flow-instance* $\Rightarrow$
($'commodity \Rightarrow 'route \Rightarrow real) \Rightarrow bool$ **where**
fractional-feasible I f $\longleftrightarrow$
well-formed-instance I $\wedge$
$(\forall k\ r.\ 0 \le f\ k\ r) \wedge$
$(\forall k\ r.\ \neg\ \textit{pf-admissible I k r} \longrightarrow f\ k\ r = 0) \wedge$
$(\forall k.\ (\sum r\in UNIV.\ f\ k\ r) = \textit{pf-demand I k}) \wedge$
$(\forall a.\ \textit{fractional-load I f a} \le \textit{pf-capacity I a})$

definition *unsplittable-routing* ::
($'commodity$, $'route$, $'arc$) *path-flow-instance* $\Rightarrow$
($'commodity \Rightarrow 'route) \Rightarrow bool$ **where**
unsplittable-routing I q $\longleftrightarrow$ $(\forall k.\ \textit{pf-admissible I k}\ (q\ k))$

definition *unsplittable-load* ::
  ('*commodity*::*finite*, '*route*, '*arc*) *path-flow-instance* ⇒
   ('*commodity* ⇒ '*route*) ⇒ '*arc* ⇒ *real* **where**
 unsplittable-load I q a =
   ($\sum$ *k*∈*UNIV*. *if pf-uses I k* (*q k*) *a then pf-demand I k else 0*)

definition *unsplittable-cost* ::
  ('*commodity*::*finite*, '*route*, '*arc*::*finite*) *path-flow-instance* ⇒
   ('*commodity* ⇒ '*route*) ⇒ *real* **where**
 unsplittable-cost I q =
   ($\sum$ *a*∈*UNIV*. *pf-cost I a* ∗ *unsplittable-load I q a*)

definition *weak-dgg-rounding* ::
  ('*commodity*::*finite*, '*route*::*finite*, '*arc*::*finite*) *path-flow-instance* ⇒
   ('*commodity* ⇒ '*route* ⇒ *real*) ⇒
   ('*commodity* ⇒ '*route*) ⇒ *bool* **where**
 weak-dgg-rounding I f q ⟷
   unsplittable-routing I q ∧
   (∀ *a*. *unsplittable-load I q a*
     ≤ *fractional-load I f a* + *maximum-demand I*) ∧
   unsplittable-cost I q ≤ *fractional-cost I f*

definition *strict-dgg-rounding* ::
  ('*commodity*::*finite*, '*route*::*finite*, '*arc*::*finite*) *path-flow-instance* ⇒
   ('*commodity* ⇒ '*route* ⇒ *real*) ⇒
   ('*commodity* ⇒ '*route*) ⇒ *bool* **where**
 strict-dgg-rounding I f q ⟷
   unsplittable-routing I q ∧
   (∀ *a*. *unsplittable-load I q a*
     < *fractional-load I f a* + *maximum-demand I*) ∧
   unsplittable-cost I q ≤ *fractional-cost I f*

definition *weak-dgg-property* ::
  ('*commodity*::*finite*, '*route*::*finite*, '*arc*::*finite*) *path-flow-instance* ⇒
   ('*commodity* ⇒ '*route* ⇒ *real*) ⇒ *bool* **where**
 weak-dgg-property I f ⟷ (∃ *q*. *weak-dgg-rounding I f q*)

definition *strict-dgg-property* ::
  ('*commodity*::*finite*, '*route*::*finite*, '*arc*::*finite*) *path-flow-instance* ⇒
   ('*commodity* ⇒ '*route* ⇒ *real*) ⇒ *bool* **where**
 strict-dgg-property I f ⟷ (∃ *q*. *strict-dgg-rounding I f q*)

definition *dgg-counterexample* ::
  ('*commodity*::*finite*, '*route*::*finite*, '*arc*::*finite*) *path-flow-instance* ⇒
   ('*commodity* ⇒ '*route* ⇒ *real*) ⇒ *bool* **where**
 dgg-counterexample I f ⟷
   well-formed-instance I ∧ *fractional-feasible I f* ∧
   ¬ *weak-dgg-property I f*

lemma *strict-dgg-rounding-imp-weak*:
strict-dgg-rounding I f q $\Longrightarrow$ *weak-dgg-rounding I f q*
⟨*proof*⟩

2 The directed graph and all of its terminal paths

datatype *vertex* =
Source
| *Junction-U*
| *Junction-V*
| *Junction-W*
| *Terminal-1*
| *Terminal-2*
| *Terminal-3*

datatype *commodity* = *Commodity-1* | *Commodity-2* | *Commodity-3*

datatype *arc* =
S-T1
| *S-T2*
| *S-U*
| *U-V*
| *U-T3*
| *V-T1*
| *V-W*
| *W-T2*
| *W-T3*

datatype *route-choice* = *Paid* | *Free*

fun *arc-tail* :: *arc* ⇒ *vertex* **where**
arc-tail S-T1 = *Source*
| *arc-tail S-T2* = *Source*
| *arc-tail S-U* = *Source*
| *arc-tail U-V* = *Junction-U*
| *arc-tail U-T3* = *Junction-U*
| *arc-tail V-T1* = *Junction-V*
| *arc-tail V-W* = *Junction-V*
| *arc-tail W-T2* = *Junction-W*
| *arc-tail W-T3* = *Junction-W*

fun *arc-head* :: *arc* ⇒ *vertex* **where**
arc-head S-T1 = *Terminal-1*
| *arc-head S-T2* = *Terminal-2*
| *arc-head S-U* = *Junction-U*
| *arc-head U-V* = *Junction-V*
| *arc-head U-T3* = *Terminal-3*
| *arc-head V-T1* = *Terminal-1*
| *arc-head V-W* = *Junction-W*

| *arc-head W-T2* = *Terminal-2*
| *arc-head W-T3* = *Terminal-3*

fun *commodity-terminal* :: *commodity* ⇒ *vertex* **where**
commodity-terminal Commodity-1 = *Terminal-1*
| *commodity-terminal Commodity-2* = *Terminal-2*
| *commodity-terminal Commodity-3* = *Terminal-3*

fun *edge-path* :: *vertex* ⇒ *arc list* ⇒ *vertex* ⇒ *bool* **where**
edge-path u [] *v* = (*u* = *v*)
| *edge-path u* (*a* # *as*) *v* =
(*arc-tail a* = *u* ∧ *edge-path* (*arc-head a*) *as v*)

lemma *arc-tail-eq-source* [*simp*]:
arc-tail a = *Source* ⟷ *a* = *S-T1* ∨ *a* = *S-T2* ∨ *a* = *S-U*
⟨*proof*⟩

lemma *arc-tail-eq-junction-u* [*simp*]:
arc-tail a = *Junction-U* ⟷ *a* = *U-V* ∨ *a* = *U-T3*
⟨*proof*⟩

lemma *arc-tail-eq-junction-v* [*simp*]:
arc-tail a = *Junction-V* ⟷ *a* = *V-T1* ∨ *a* = *V-W*
⟨*proof*⟩

lemma *arc-tail-eq-junction-w* [*simp*]:
arc-tail a = *Junction-W* ⟷ *a* = *W-T2* ∨ *a* = *W-T3*
⟨*proof*⟩

lemma *arc-tail-eq-terminal-1* [*simp*]:
arc-tail a = *Terminal-1* ⟷ *False*
⟨*proof*⟩

lemma *arc-tail-eq-terminal-2* [*simp*]:
arc-tail a = *Terminal-2* ⟷ *False*
⟨*proof*⟩

lemma *arc-tail-eq-terminal-3* [*simp*]:
arc-tail a = *Terminal-3* ⟷ *False*
⟨*proof*⟩

lemma *edge-path-from-terminal-1* [*simp*]:
edge-path Terminal-1 as v ⟷ *as* = [] ∧ *v* = *Terminal-1*
⟨*proof*⟩

lemma *edge-path-from-terminal-2* [*simp*]:
edge-path Terminal-2 as v ⟷ *as* = [] ∧ *v* = *Terminal-2*
⟨*proof*⟩

lemma *edge-path-from-terminal-3* [*simp*]:
  edge-path Terminal-3 as v $\longleftrightarrow$ *as* = [] $\wedge$ *v* = *Terminal-3*
  $\langle proof \rangle$

lemma *edge-path-from-junction-w* [*simp*]:
  edge-path Junction-W as v $\longleftrightarrow$
    (*as* = [] $\wedge$ *v* = *Junction-W*) $\vee$
    (*as* = [*W-T2*] $\wedge$ *v* = *Terminal-2*) $\vee$
    (*as* = [*W-T3*] $\wedge$ *v* = *Terminal-3*)
  $\langle proof \rangle$

lemma *edge-path-from-junction-v* [*simp*]:
  edge-path Junction-V as v $\longleftrightarrow$
    (*as* = [] $\wedge$ *v* = *Junction-V*) $\vee$
    (*as* = [*V-T1*] $\wedge$ *v* = *Terminal-1*) $\vee$
    (*as* = [*V-W*] $\wedge$ *v* = *Junction-W*) $\vee$
    (*as* = [*V-W*, *W-T2*] $\wedge$ *v* = *Terminal-2*) $\vee$
    (*as* = [*V-W*, *W-T3*] $\wedge$ *v* = *Terminal-3*)
  $\langle proof \rangle$

lemma *edge-path-from-junction-u* [*simp*]:
  edge-path Junction-U as v $\longleftrightarrow$
    (*as* = [] $\wedge$ *v* = *Junction-U*) $\vee$
    (*as* = [*U-V*] $\wedge$ *v* = *Junction-V*) $\vee$
    (*as* = [*U-T3*] $\wedge$ *v* = *Terminal-3*) $\vee$
    (*as* = [*U-V*, *V-T1*] $\wedge$ *v* = *Terminal-1*) $\vee$
    (*as* = [*U-V*, *V-W*] $\wedge$ *v* = *Junction-W*) $\vee$
    (*as* = [*U-V*, *V-W*, *W-T2*] $\wedge$ *v* = *Terminal-2*) $\vee$
    (*as* = [*U-V*, *V-W*, *W-T3*] $\wedge$ *v* = *Terminal-3*)
  $\langle proof \rangle$

lemma *edge-path-from-source* [*simp*]:
  edge-path Source as v $\longleftrightarrow$
    (*as* = [] $\wedge$ *v* = *Source*) $\vee$
    (*as* = [*S-T1*] $\wedge$ *v* = *Terminal-1*) $\vee$
    (*as* = [*S-T2*] $\wedge$ *v* = *Terminal-2*) $\vee$
    (*as* = [*S-U*] $\wedge$ *v* = *Junction-U*) $\vee$
    (*as* = [*S-U*, *U-V*] $\wedge$ *v* = *Junction-V*) $\vee$
    (*as* = [*S-U*, *U-T3*] $\wedge$ *v* = *Terminal-3*) $\vee$
    (*as* = [*S-U*, *U-V*, *V-T1*] $\wedge$ *v* = *Terminal-1*) $\vee$
    (*as* = [*S-U*, *U-V*, *V-W*] $\wedge$ *v* = *Junction-W*) $\vee$
    (*as* = [*S-U*, *U-V*, *V-W*, *W-T2*] $\wedge$ *v* = *Terminal-2*) $\vee$
    (*as* = [*S-U*, *U-V*, *V-W*, *W-T3*] $\wedge$ *v* = *Terminal-3*)
$\langle proof \rangle$

fun *route-edges* :: *commodity* $\Rightarrow$ *route-choice* $\Rightarrow$ *arc list* **where**
  route-edges Commodity-1 Paid = [*S-T1*]
| *route-edges Commodity-1 Free* = [*S-U*, *U-V*, *V-T1*]
| *route-edges Commodity-2 Paid* = [*S-T2*]

| *route-edges Commodity-2 Free* = [*S-U*, *U-V*, *V-W*, *W-T2*]
| *route-edges Commodity-3 Paid* = [*S-U*, *U-T3*]
| *route-edges Commodity-3 Free* = [*S-U*, *U-V*, *V-W*, *W-T3*]

lemma *route-edges-valid*:
edge-path Source (*route-edges k r*) (*commodity-terminal k*)
⟨*proof*⟩

theorem *all-source-terminal-paths*:
edge-path Source as (*commodity-terminal k*) ⟷
as = *route-edges k Paid* ∨ *as* = *route-edges k Free*
⟨*proof*⟩

corollary *graph-paths-exactly-routes*:
{*as*. *edge-path Source as* (*commodity-terminal k*)} =
{*route-edges k Paid*, *route-edges k Free*}
⟨*proof*⟩

3 The counterexample instance

fun *demand-amount* :: *commodity* ⇒ *real* **where**
demand-amount Commodity-1 = *15*
| *demand-amount Commodity-2* = *10*
| *demand-amount Commodity-3* = *15*

fun *arc-capacity* :: *arc* ⇒ *real* **where**
arc-capacity S-T1 = *10*
| *arc-capacity S-T2* = *6*
| *arc-capacity S-U* = *24*
| *arc-capacity U-V* = *14*
| *arc-capacity U-T3* = *10*
| *arc-capacity V-T1* = *5*
| *arc-capacity V-W* = *9*
| *arc-capacity W-T2* = *4*
| *arc-capacity W-T3* = *5*

fun *arc-cost* :: *arc* ⇒ *real* **where**
arc-cost S-T1 = *2*
| *arc-cost S-T2* = *3*
| *arc-cost S-U* = *0*
| *arc-cost U-V* = *0*
| *arc-cost U-T3* = *2*
| *arc-cost V-T1* = *0*
| *arc-cost V-W* = *0*
| *arc-cost W-T2* = *0*
| *arc-cost W-T3* = *0*

definition *counterexample-uses* ::
commodity ⇒ *route-choice* ⇒ *arc* ⇒ *bool* **where**

```
  counterexample-uses k r a ⟷ a ∈ set (route-edges k r)

definition counterexample-instance ::
    (commodity, route-choice, arc) path-flow-instance where
  counterexample-instance =
    (|pf-admissible = (λ- -. True),
     pf-demand = demand-amount,
     pf-capacity = arc-capacity,
     pf-cost = arc-cost,
     pf-uses = counterexample-uses|)

fun split-flow :: commodity ⇒ route-choice ⇒ real where
  split-flow Commodity-1 Paid = 10
| split-flow Commodity-1 Free = 5
| split-flow Commodity-2 Paid = 6
| split-flow Commodity-2 Free = 4
| split-flow Commodity-3 Paid = 10
| split-flow Commodity-3 Free = 5

lemma UNIV-commodity [simp]:
  (UNIV :: commodity set) =
   {Commodity-1, Commodity-2, Commodity-3}
  ⟨proof⟩

lemma UNIV-route-choice [simp]:
  (UNIV :: route-choice set) = {Paid, Free}
  ⟨proof⟩

lemma UNIV-arc [simp]:
  (UNIV :: arc set) =
   {S-T1, S-T2, S-U, U-V, U-T3, V-T1, V-W, W-T2, W-T3}
  ⟨proof⟩

instantiation commodity :: finite
begin

instance
⟨proof⟩

end

instantiation route-choice :: finite
begin

instance
⟨proof⟩

end
```

instantiation *arc* :: *finite*
begin

instance
$\langle proof \rangle$

end

lemma *counterexample-well-formed*:
well-formed-instance counterexample-instance
$\langle proof \rangle$

lemma *counterexample-maximum-demand* [*simp*]:
maximum-demand counterexample-instance = *15*
$\langle proof \rangle$

lemma *split-flow-meets-demands*:
$(\sum r \in UNIV.\ split\text{-}flow\ k\ r) = demand\text{-}amount\ k$
$\langle proof \rangle$

lemma *split-flow-load* [*simp*]:
fractional-load counterexample-instance split-flow a = *arc-capacity a*
$\langle proof \rangle$

theorem *split-flow-saturates-capacities*:
$\forall a.\ fractional\text{-}load\ counterexample\text{-}instance\ split\text{-}flow\ a =$
$pf\text{-}capacity\ counterexample\text{-}instance\ a$
$\langle proof \rangle$

lemma *split-flow-bound-iff-capacity-bound*:
$(\forall a.\ unsplittable\text{-}load\ counterexample\text{-}instance\ q\ a$
$\le fractional\text{-}load\ counterexample\text{-}instance\ split\text{-}flow\ a + 15)$
$\longleftrightarrow$
$(\forall a.\ unsplittable\text{-}load\ counterexample\text{-}instance\ q\ a$
$\le pf\text{-}capacity\ counterexample\text{-}instance\ a + 15)$
$\langle proof \rangle$

theorem *split-flow-feasible*:
fractional-feasible counterexample-instance split-flow
$\langle proof \rangle$

theorem *split-flow-cost*:
fractional-cost counterexample-instance split-flow = *58*
$\langle proof \rangle$

4 Every bounded unsplittable routing costs at least sixty

Every paid route costs exactly thirty. If at most one commodity takes its paid route, one of three backbone arcs violates the additive bound: the load on S-U is forty when only commodity three is paid, the load on U-V is thirty when only commodity two is paid, and the load on V-W is twenty-five when only commodity one is paid. Their respective allowed loads are thirty-nine, twenty-nine, and twenty-four.

definition *paid-count* :: (*commodity* ⇒ *route-choice*) ⇒ *nat* **where**
paid-count q =
(*if q Commodity-1* = *Paid then 1 else 0*) +
(*if q Commodity-2* = *Paid then 1 else 0*) +
(*if q Commodity-3* = *Paid then 1 else 0*)

lemma *unsplittable-load-s-u*:
unsplittable-load counterexample-instance q S-U =
(*if q Commodity-1* = *Free then 15 else 0*) +
(*if q Commodity-2* = *Free then 10 else 0*) + *15*
⟨*proof*⟩

lemma *unsplittable-load-u-v*:
unsplittable-load counterexample-instance q U-V =
(*if q Commodity-1* = *Free then 15 else 0*) +
(*if q Commodity-2* = *Free then 10 else 0*) +
(*if q Commodity-3* = *Free then 15 else 0*)
⟨*proof*⟩

lemma *unsplittable-load-v-w*:
unsplittable-load counterexample-instance q V-W =
(*if q Commodity-2* = *Free then 10 else 0*) +
(*if q Commodity-3* = *Free then 15 else 0*)
⟨*proof*⟩

lemma *bounded-routing-has-two-paid*:
assumes *bound*:
∀ *a. unsplittable-load counterexample-instance q a*
≤ *fractional-load counterexample-instance split-flow a* + *15*
shows *2* ≤ *paid-count q*
⟨*proof*⟩

lemma *unsplittable-cost-eq-paid-count*:
unsplittable-cost counterexample-instance q = *30* ∗ *real* (*paid-count q*)
⟨*proof*⟩

theorem *bounded-unsplittable-cost*:
assumes *bound*:
∀ *a. unsplittable-load counterexample-instance q a*

$\leq$ *fractional-load counterexample-instance split-flow a + 15*
shows *60* $\leq$ *unsplittable-cost counterexample-instance q*
$\langle proof \rangle$

corollary *capacity-bounded-unsplittable-cost*:
assumes *bound*:
$\forall$ *a. unsplittable-load counterexample-instance q a*
$\leq$ *pf-capacity counterexample-instance a + 15*
shows *60* $\leq$ *unsplittable-cost counterexample-instance q*
$\langle proof \rangle$

theorem *no-weak-dgg-rounding*:
$\neg$ *weak-dgg-property counterexample-instance split-flow*
$\langle proof \rangle$

corollary *no-strict-dgg-rounding*:
$\neg$ *strict-dgg-property counterexample-instance split-flow*
$\langle proof \rangle$

theorem *dinitz-garg-goemans-cost-counterexample*:
dgg-counterexample counterexample-instance split-flow
$\langle proof \rangle$

corollary *weak-dgg-cost-conjecture-false*:
$\neg$ ($\forall$ (*I* :: (*commodity*, *route-choice*, *arc*) *path-flow-instance*) *f*.
well-formed-instance I $\wedge$ *fractional-feasible I f*
$\longrightarrow$ *weak-dgg-property I f*)
$\langle proof \rangle$

corollary *strict-dgg-cost-conjecture-false*:
$\neg$ ($\forall$ (*I* :: (*commodity*, *route-choice*, *arc*) *path-flow-instance*) *f*.
well-formed-instance I $\wedge$ *fractional-feasible I f*
$\longrightarrow$ *strict-dgg-property I f*)
$\langle proof \rangle$

end

References

[1] Y. Dinitz, N. Garg, and M. X. Goemans. On the single-source unsplittable flow problem. *Combinatorica*, 19(1):17–41, 1999. DOI: `https://doi.org/10.1007/s004930050043`.

[2] D. Rybin. Counterexample to the Dinitz–Garg–Goemans cost conjecture. Post on X, July 2026. `https://x.com/DmitryRybin1`, posted July 22, 2026.

[3] V. Traub, L. Vargas Koch, and R. Zenklusen. Single-source unsplittable flows in planar and bounded-genus graphs. *Mathematical Programming*, 2026. Published online July 22, 2026. DOI: `https://doi.org/10.1007/s10107-026-02365-x`.