matfyz

Computer Graphics Charles University

Trojúhelníkové sítě

© 2009-2015 Josef Pelikán, CGG MFF UK Praha

http://cgg.mff.cuni.cz/~pepca/

pepca@cgg.mff.cuni.cz

Trojúhelníkové sítě (tri-mesh)

- úsporné reprezentace v paměti
- komprese
  - ukládání na server, přenos po síti
- zjednodušování sítí a „Level of Detail“ (LoD)
  - statické, dynamické, diskrétní i spojité

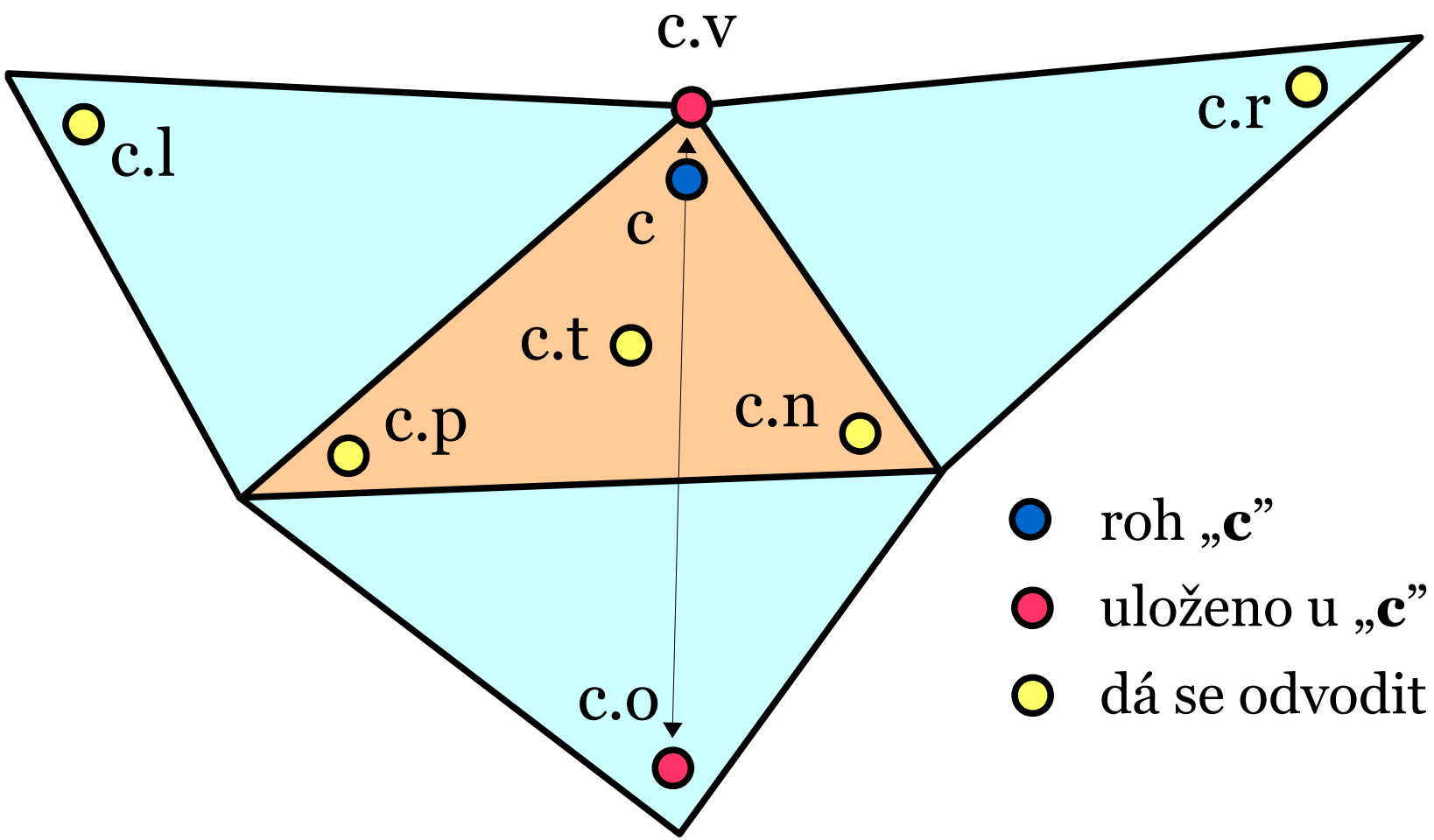
Reprezentace tri-mesh

- podmíněné **HW zobrazováním**
  - pole vrcholů („vertex-array“), „index-array“
  - chybí topologie (informace o sousedech)
- klasická **topologické rozšíření**
  - okřídlená hrana („winged-edge“), půlhrana („half-edge“)
  - implementace s ukazateli je paměťově náročná
- úsporné reprezentace
  - k poli vrcholů přidávají jen minimální informaci
  - např. „**Corner Table**“

„Corner Table“

- tabulka vrcholů G[v]
  - souřadnice, normála, barva, texturové souřadnice, …
- tabulka rohů V[c]
  - jeden vnitřní roh trojúhelníka
  - **index vrcholu** („**c.v**“)
  - rohy jsou uloženy za sebou v 1D poli, CW orientace stěn
  - **protější roh protějšího trojúhelníka** („**c.o**“)
  - implicitní údaje:
    - číslo trojúhelníka **t** = c div 3
    - ostatní rohy **c.n** = (c mod 3 == 2) ? c-2 : c+1, **c.p** = c.n.n
    - další sousední trojúhelníky **c.l** = c.n.o, **c.r** = c.p.o

„Corner Table“



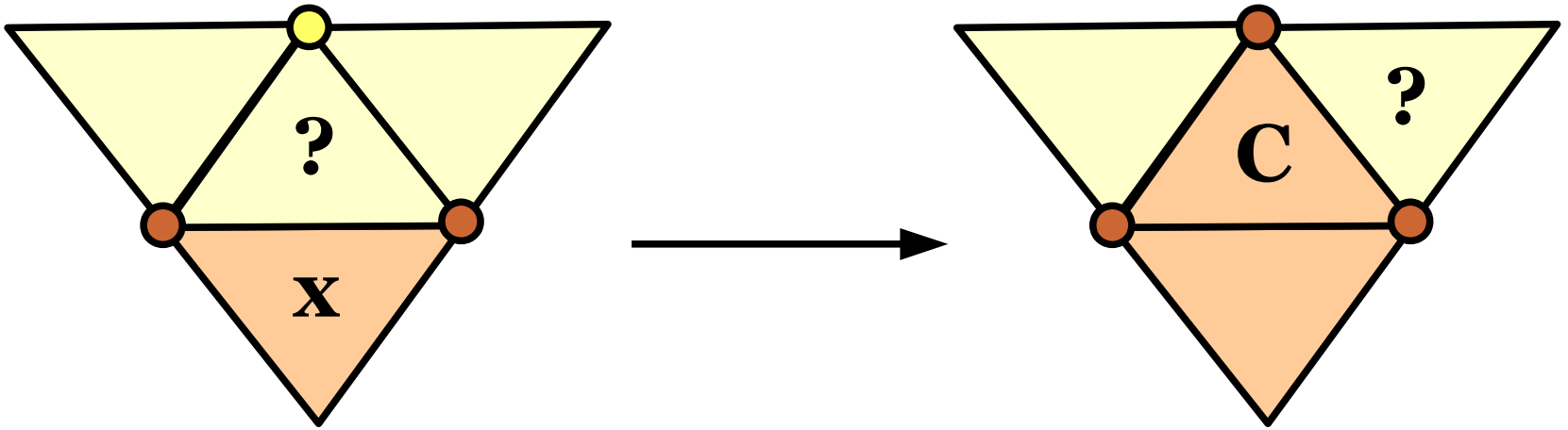
„Corner Table“ – přenos

- tabulka „**o**“ se nemusí přenášet
  - lze ji jednoznačně rekonstruovat z „**v**“
- hodnoty indexů „**v**“ se mohou bitově ořezat
  - index vrcholu obsahuje $31 - \log_2 v$ úvodních nul
- ořezání souřadnic
  - obalový kvádr celého objektu, uvnitř se používají relativní souřadnice (10 až 16 bitů na složku)
  - predikce polohy vrcholů – při inkrementálním průchodu daty (např. „Edgebreaker“) – další úspory

„Edgebreaker“ (Rossignac, 1999)

- úsporné kódování tri-mesh pomocí inkrementálního průchodu sítí
  - **pozice vrcholů** se komprimují za pomoci predikce (až 7 bpv)
  - **topologie sítě** se ukládá velice úsporně na základě průchodu (1.0 až 1.8 bpv)
- „CLERS“ pole
  - **5 možností**, jak pokračovat z aktuálního trojúhelníka
  - možnost entropické komprese (některé kroky/ posloupnosti jsou mnohem pravděpodobnější)

CLERS kódování – C



x předchozí trojúhelník

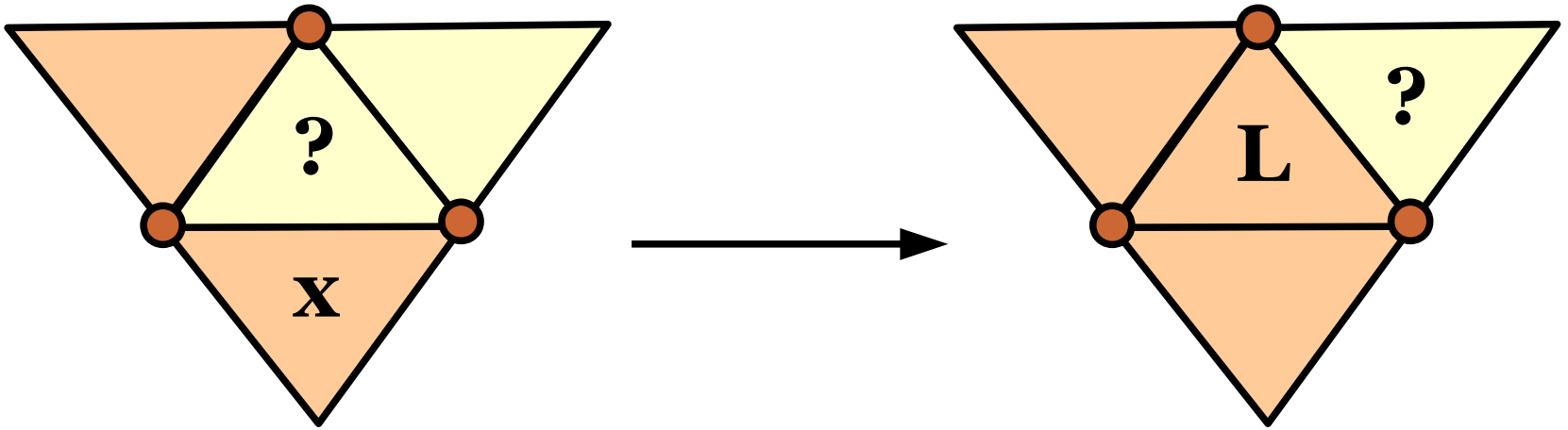
? aktuální trojúhelník

navštívený vrchol

nenavštívený vrchol

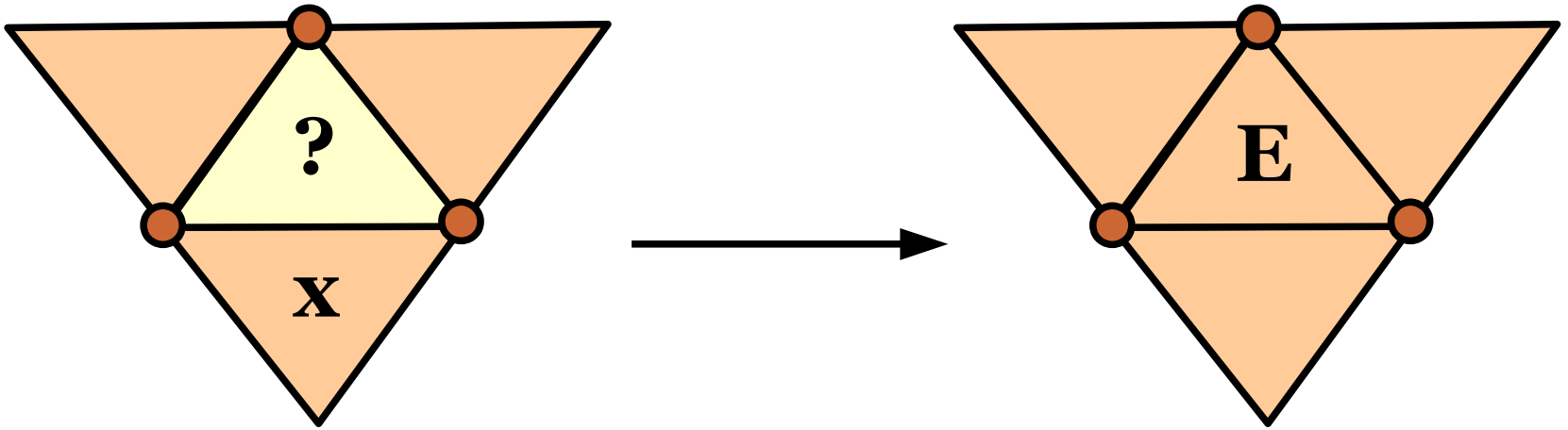
```
write( "C" );
c = c.r;
```

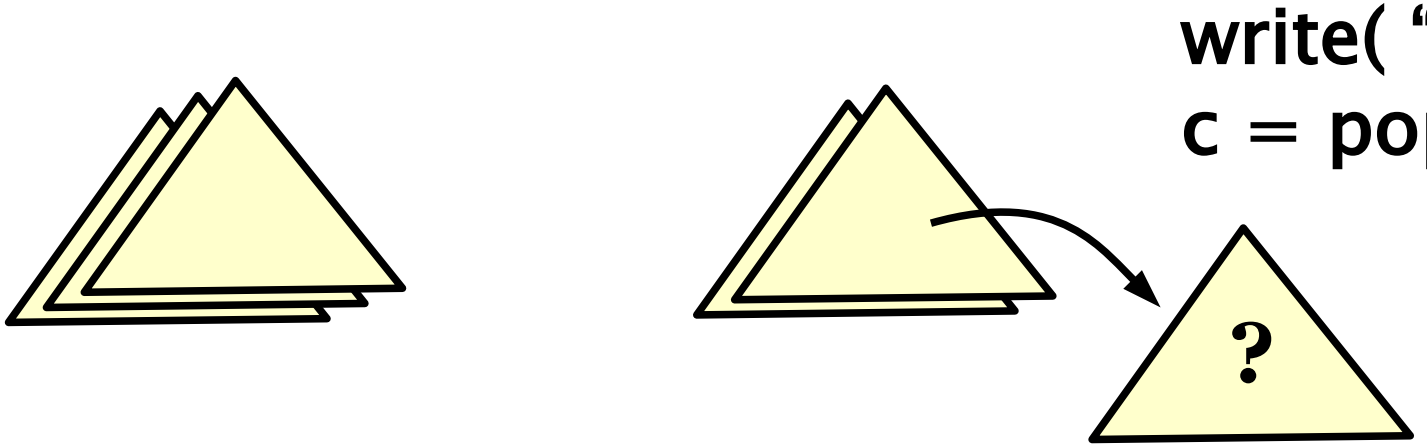
CLERS kódování – L



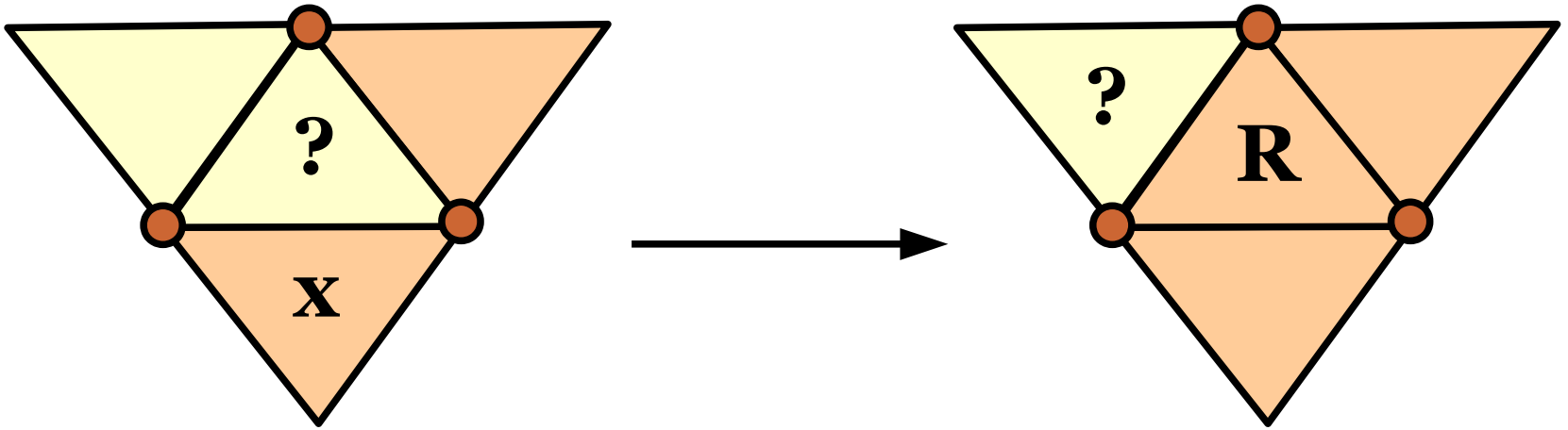

```
write( “L” );
c = c.r;
```

CLERS kódování – E



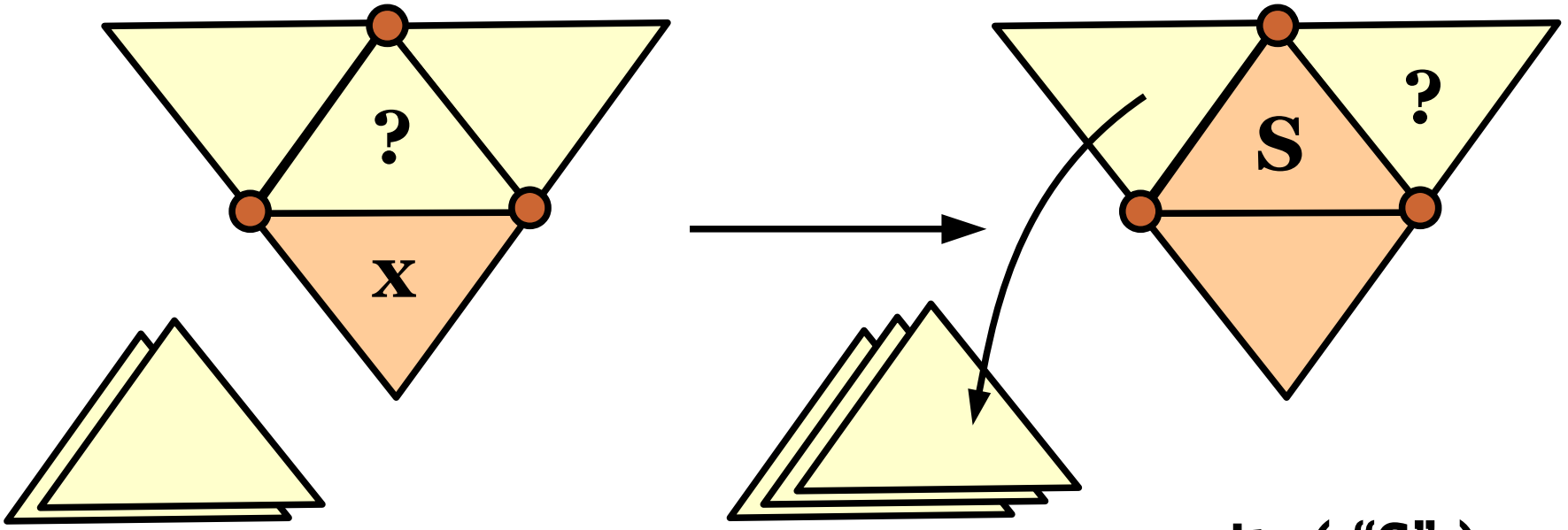


CLERS kódování – R




```
write( “R” );
c = c.l;
```

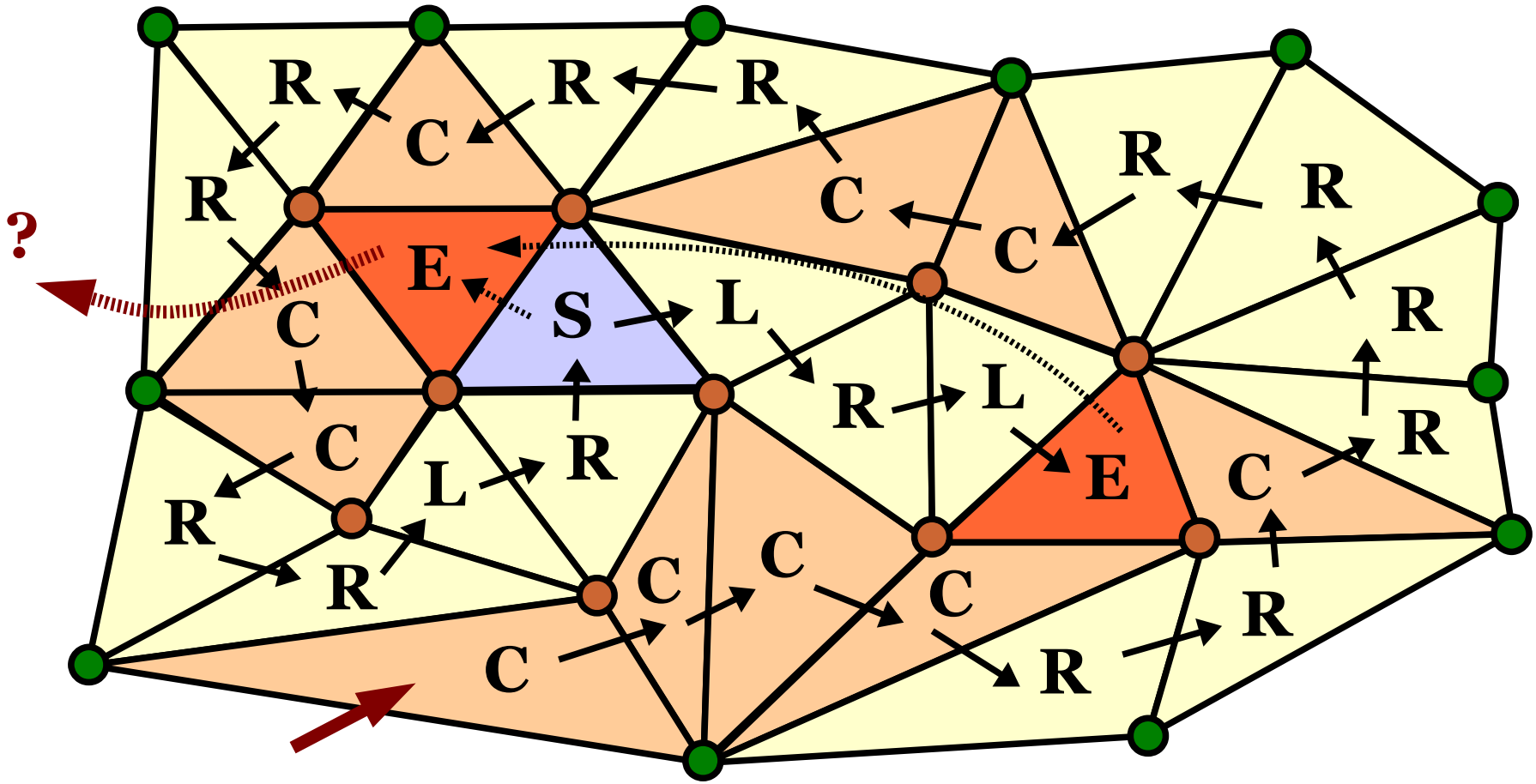
CLERS kódování – S




```
write( “S” );
push( c.l );
c = c.r;
```

Průchod sítí – příklad

(obvod oblasti již byl navštíven ●)



CCCCRRCRRRRRCCRRCRRCRRCCRRLRSLRLEE..

Kódování průchodu

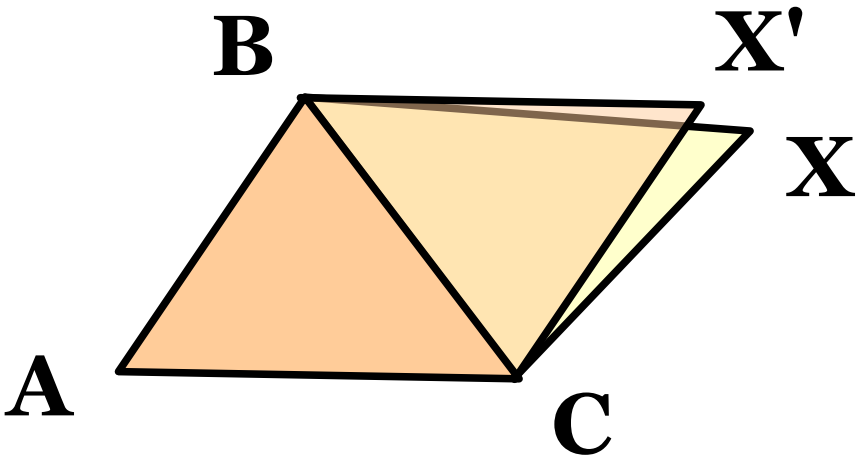
- jedině krok „**C**“ navštěvuje nový vrchol
  - tj. přibližně **½ kroků** musí být typu „**C**“
  - statický kód: **C=0**, **L=110**, **E=111**, **R=101**, **S=100**
  - amortizovaná délka kódu bude maximálně **2t**
  - příklad: 10·1 + 20·3 = 70 bitů na 30 trojúhelníků
- efektivnější statický kód (< 1.84t)
  - kroky následující za „**C**“: **C=0**, **S=10**, **R=11**
  - bez předcházejícího „**C**“: existují 3 kódy, z nichž vždy aspoň jeden dosahuje délky **< 11t/6**
  - jeden z nich: **C=00**, **L=110**, **E=01**, **R=10**, **S=111**

Kódování průchodu

- další metoda (≤ **2t**, průměrně mezi **1.3t** a **1.6t**)
  - dvojice kroků: **CR=01**, **CC=00**, **CS=1101**
  - jednotlivé kroky: **L=1110**, **E=1100**, **R=10**, **S=1111**
  - příklad: (00)(00)(10)(10)(01)(10)(10)(10)(00)(10)(10)(01)(10)(00)(10)(10)(1110)(10)(1111)(1110)(10)(1110)(1100)(1100) … 60 bitů na 30 trojúhelníků (10 vrcholů)
- dynamické entropické kódování CLERS řetězce
  - např. Huffmanův kód (včetně přenosu Huff. stromu) – redukce až na **< 1.0t**
  - dynamické kódování (aritmetický kód) může být ještě efektivnější

Kódování geometrie

- predikce z předchozího navštíveného vrcholu
  - nebo z obou již navštívených vrcholů téhož trojúhelníka
  - očekávaná úspora: ½ bitů pro reprezentaci polohy
- predikce ze sousedního trojúhelníka
  - rovnoběžník: $\mathbf{X} \approx \mathbf{X'} = \mathbf{B} + \mathbf{C} - \mathbf{A}$



Kódování geometrie

- predikce úhlu mezi dvěma sousedními trojúhelníky
  - může ještě dále vylepšit rovnoběžníkové pravidlo
  - predikce ze sousedních trojúhelníků nebo z globální statistiky sítě
- predikce z několika předchozích trojúhelníků
  - čtyři spojití předchůdci
  - lineární koeficienty predikce se optimalizují a přenášejí v hlavičce souboru
- v příznivých případech se dosahuje až **7 bpv** pro kódování geometrie

Úroveň detailu (LoD)

Optimální **efektivita vykreslování**:

- **vzdálené detaily** (velikostí srovnatelné s rozměrem pixelu) se už nemusí vykreslovat
- naopak – **nejbližší předměty** (na které se dívá uživatel) si zasluhují co možná nejlepší vizuální kvalitu

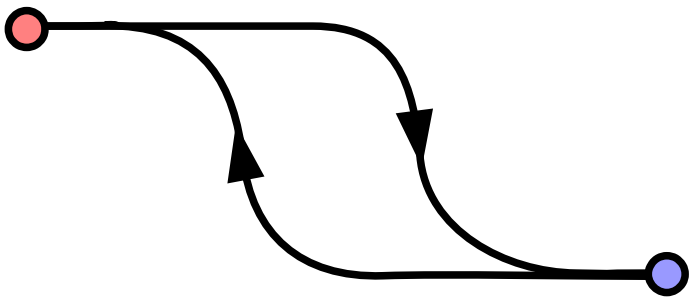
- **dynamická úroveň detailu** („dynamic Level of Detail”)
  - program automaticky přizpůsobuje jemnost dat
  - lze implementovat globální dolaďování (např. zadání celkového počtu trojúhelníků, které se mají vykreslovat)
  - náročná příprava dat: předem se musí připravit několik úrovní dat (nebo algoritmus spojitého zjemňování)

Diskrétní LoD

- **předem** je připraveno několik LoD-ů
  - modely objektu s různou přesností
  - může se vycházet z nejjemnějšího modelu – **generalizace** (časově náročný výpočet)
- **vykreslování – výběr** použitého LoDu:
  - podle vzdálenosti od kamery
  - podle velikosti projekce obalového tělesa
    - nebo přesná projekce – nejvíc pozorovatelné jsou chyby kolmé na směr pohledu
  - důležitost objektu, zaměření („focus”) pozorovatele
  - globální vyvážení (počet vykreslovaných polygonů)

Přepínání LoD

- **jednoduché přepínání**
  - musí se zavést hystereze !
  - „popping" – blikání LoDů



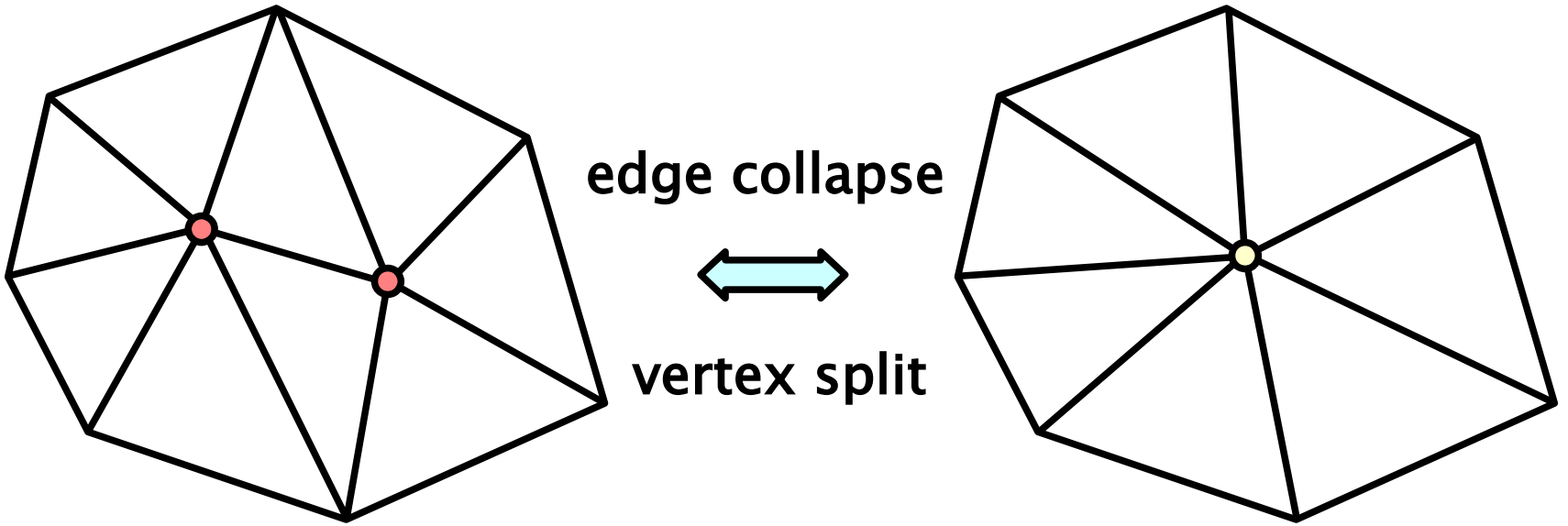
- **míchání** sousedních LoDů („LoD blending")
  - kreslení dvou sousedních úrovní pomocí **průhlednosti**
    - lineární kombinace („transition")
  - alternativa
    - aktuální LoD se kreslí neprůhledně
    - přidá se poloprůhledný nový LoD (alfa operace „over")
    - zápis „z-write" je zapnutý jen u aktuálního LoDu

Inkrementální LoD

- potlačuje blikání
  - přechod je spojitý – množství jednoduchých (elementárních) přechodů
  - bývá náročný na přípravu dat a FPU
- příklad **elementárních operací** nad sítí trojúhelníků:
  - **kolaps hrany** (a dvou sousedních trojúhelníků)
    - zmizí 1 vrchol, 3 hrany a 2 trojúhelníky
  - **rozpojení vrcholu** (inverzní operace)
    - vznikne nový vrchol, 3 hrany a 2 trojúhelníky
  - možnost animace: vrchol míří doprostřed hrany..

Elementární operace I

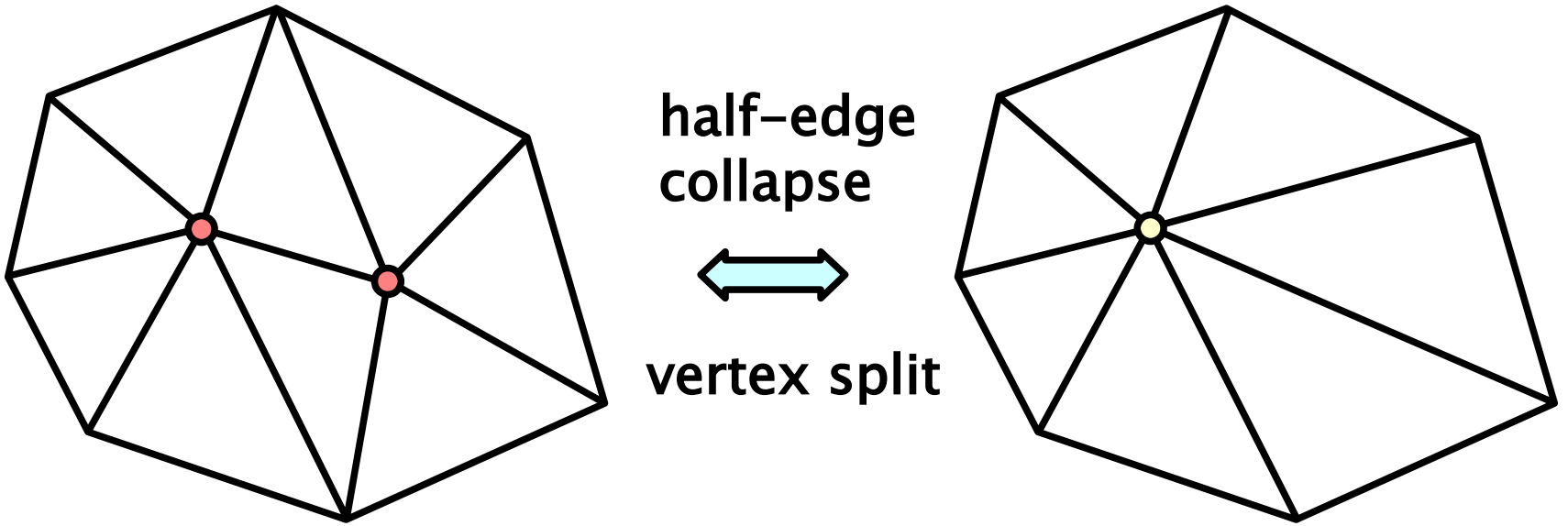
- **kolaps hrany** vs. **rozpojení vrcholu**:



- možnost přechodové animace

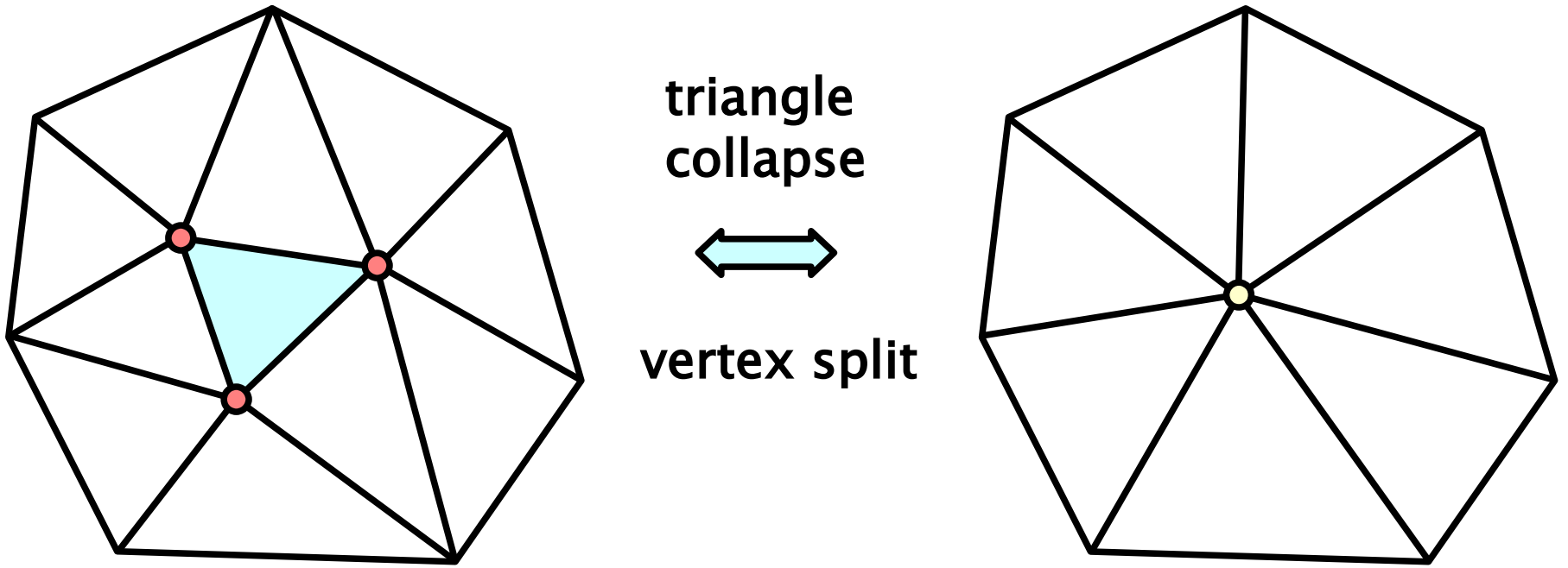
Elementární operace II

- jeden z **vrcholů** může zůstat na místě:



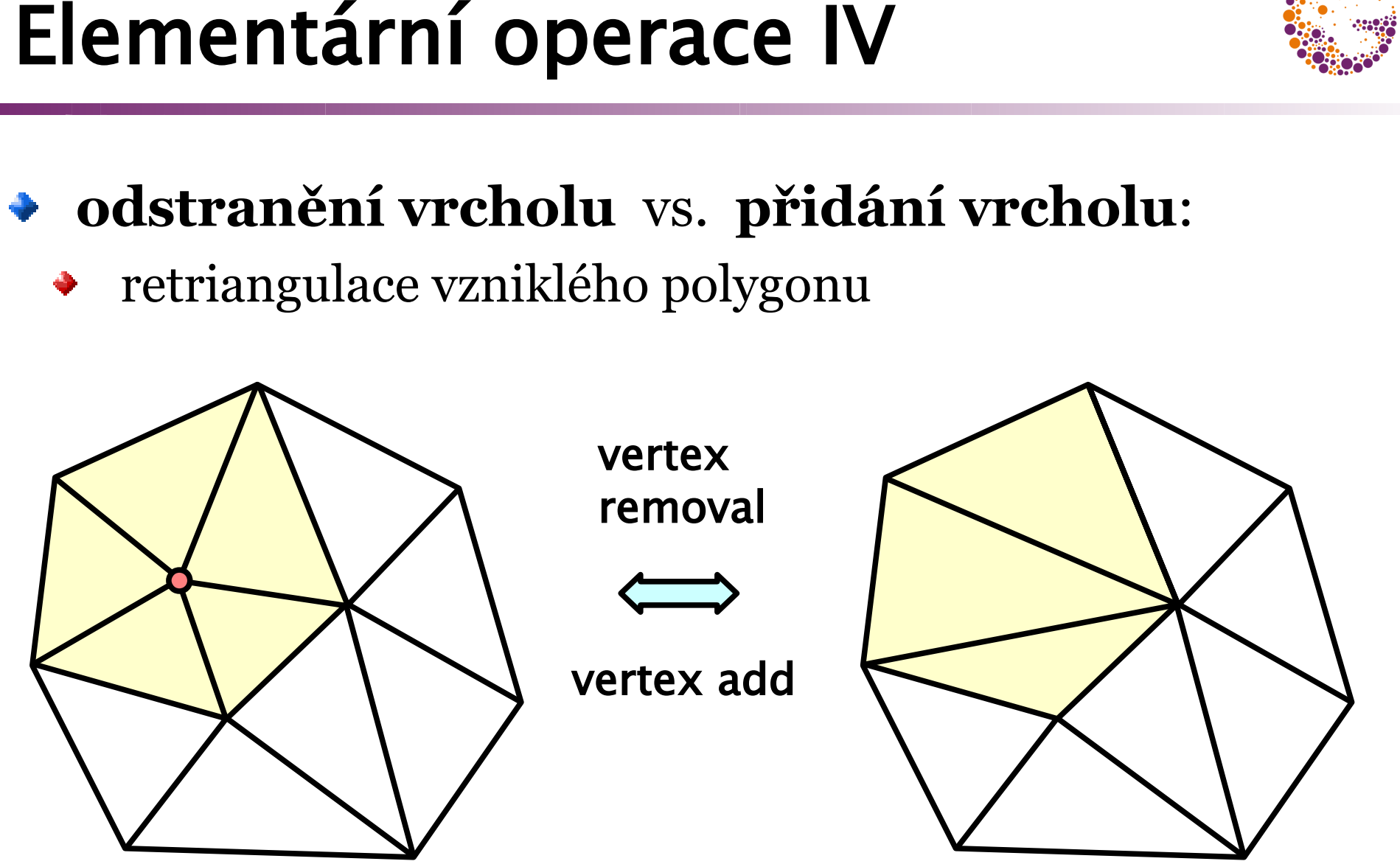
- je to vlastně jedna z forem odstranění vrcholu (viz dále)

Elementární operace III



- **kolaps trojúhelníka** vs. **rozpojení vrcholu**:

Elementární operace IV

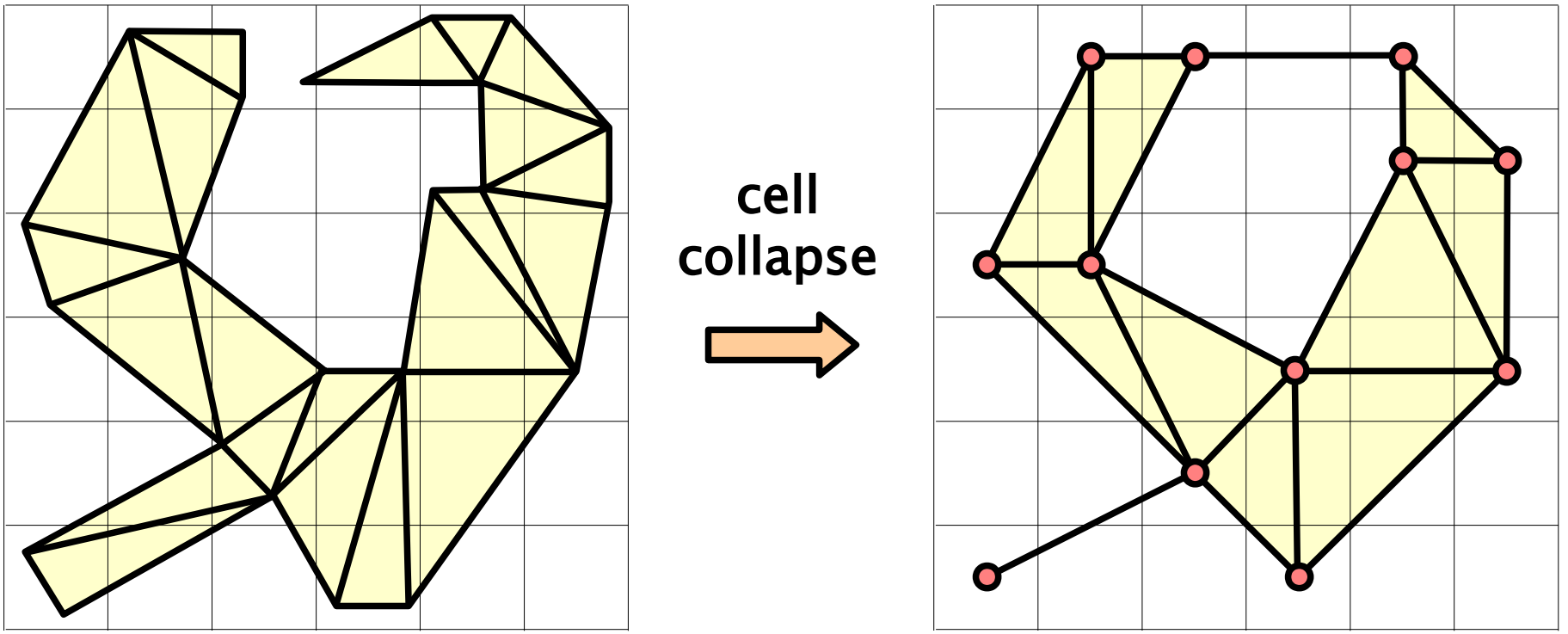


- **odstranění vrcholu** vs. **přidání vrcholu**:
  - retriangulace vzniklého polygonu

Kolapse buněk

- **síť buněk**: všechny **vrcholy ve stejné buňce** kolabují do jednoho (ve středu buňky)

  \+ indukované topologické změny



Generování LoD

- **shora dolů** (top-down)
  - základem je nejjednodušší reprezentace, přidávají se detaily
  - ve visualizaci se nepoužívá
- **zdola nahoru** (bottom-up)
  - začíná se s detailní reprezentací
  - postupné zjednodušování (redukce dat)
- různé typy **optimalizace**
  - „nejhezčí" reprezentace při daném počtu polygonů
  - globální optimalizace je NP-úplná

Optimalizace při generování LoD

- **bez optimalizace** (např. u kolapse buněk)
- **hladové přístupy** („greedy”)
  - kriteriální funkce – chybová metrika
  - vždy se uvažují všechny možnosti, jak provést zjednošující krok (hodně výpočtů metriky)
- **líné vyhodnocování** metriky („lazy evaluation”)
  - hladový přístup s redukovaným počítáním
  - po lokální změně se okolí označí jako „špinavé” („dirty”), ale zatím se pro něj metrika nepřepočítává..
  - výsledky mohou být o něco méně kvalitní (ale rychlejší)

Optimalizace při generování LoD

- **odhad metriky** („estimation”)
  - místo přesné hodnoty použiji rychlejší odhad
  - možnost kombinace s líným vyhodnocováním, tři stavy:
    - špinavý (nepřepočítáno)
    - odhad (přibližná hodnota)
    - přesná hodnota
- **nezávislé zjednodušování**
  - vybírám sadu elementárních operací, které spolu **nekolidují**, tj. mohu je provést **současně**
  - **hierarchie** – logaritmický počet úrovní (předchozí přístupy pracují lineárně)

Chybové metriky

- založené jen na **3D geometrii**
  - neuvažují se: směr pohledu, parametry projekce
- založené na **projekci** (výsledku – „target driven”)
  - efektivnější – přímo záleží na vizuální chybě výsledku
  - obrysy jsou důležitější
  - změny při mapování kontrastní textury jsou důležité
- chyba **atributu**
  - barva (často se nevhodně průměruje)
  - normálový vektor (má vliv na stínování)
  - mapování textury

Konkrétní metriky I

- vzdálenost **vrchol – vrchol**
  - pro kolapsi buněk
  - jinak lze použít jako odhad složitějších metrik
- vzdálenost **vrchol – rovina** (SP – Ronfard, 1996)
  - $\mathbf{d = p \cdot v = n_x v_x + n_y v_y + n_z v_z + D}$
  - „**supporting planes**" (**SP**): v původní síti – roviny všech stěn incidentních s daným vrcholem
  - **kolaps hrany**: SP nového vrcholu je sjednocení SP původních vrcholů
  - maximum přes všechny SP: $\mathbf{Err = max\ (p \cdot v)^2}$

Konkrétní metriky II

- **chybová kvadrika** (Garland & Heckbert, 1997)

  $\mathbf{Err = \Sigma\,(p \cdot v)^2 = \Sigma\ v^T Q_p\, v = v^T Q_{all}\, v}$

  - $\mathbf{Q_p}$ je symetrická matice 4×4 s 10 různými hodnotami spočítaná podle roviny **p**
  - kvadriky (matice) různých rovin lze **sčítat** ($\mathbf{Q_{all}}$)

- vzdálenost **vrchol – plocha**
  - složitý výpočet, může být nahrazen vzorkováním plochy

- vzdálenost **plocha – plocha**
  - různé přístupy: geometrie, textura, obalové objemy, ..

Konstantní snímková frekvence I

- elementární **zjednodušující** operace (.. edge collapse)
- elementární **zjemňující** operace (.. vertex split)

- dynamické **seznamy zjednodušení a zjemnění** (kritériem je chybová metrika)
- **seznam zjednodušení**: co už se nemusí kreslit tak přesně
  - provádění má větší prioritu (šetří vykreslovací výkon)
- **seznam zjemnění**: kde by bylo třeba kresbu zlepšit
  - musí být „časově disjunktní“ s prvním seznamem
  - při aplikaci operací se navíc kontroluje celkový objem dat

Konstantní snímková frekvence II

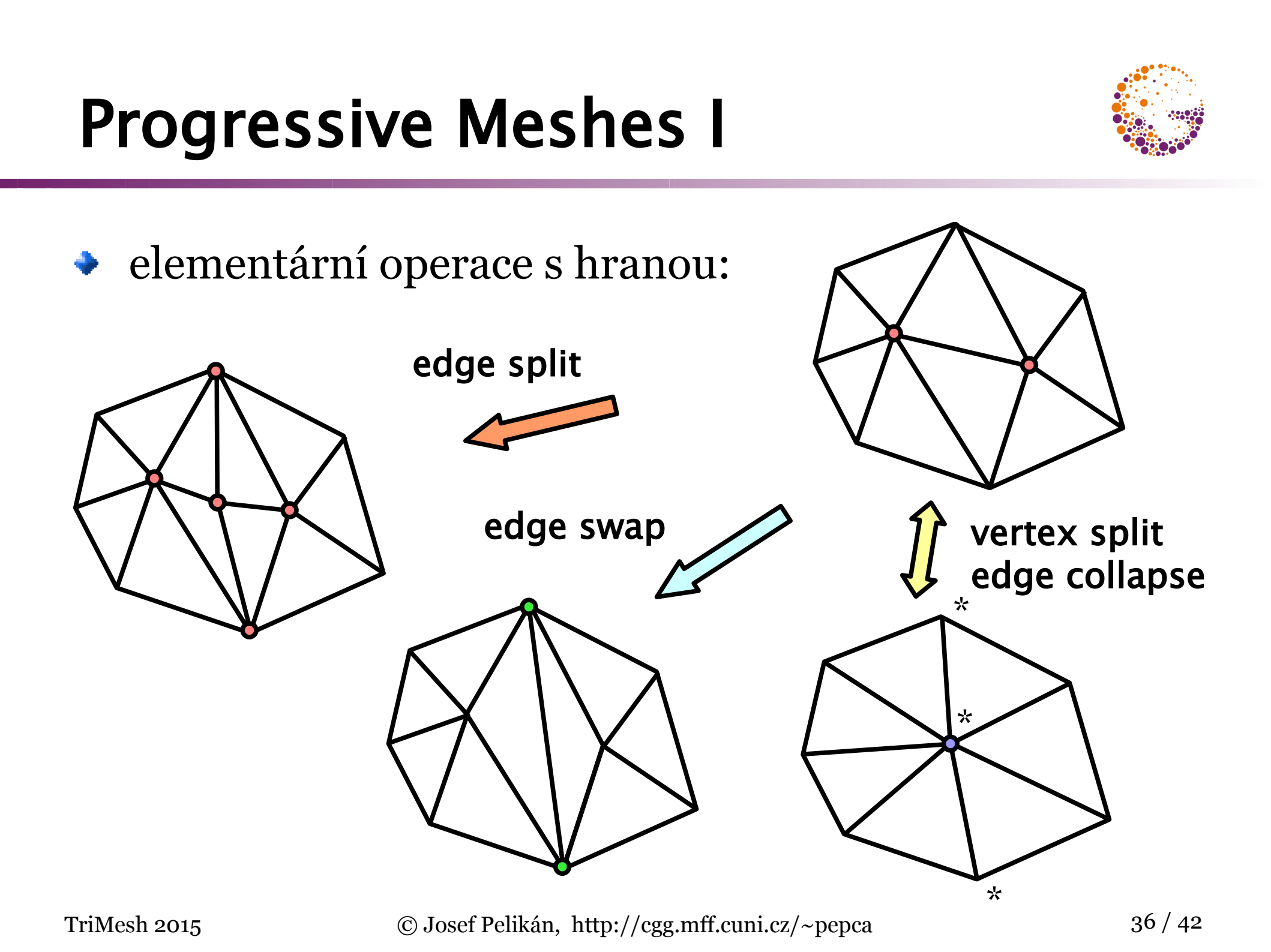
- možná implementace ve **více vláknech**:
  - první vlákno **vykresluje scénu** a případně provádí **LoD operace** (zjednodušení, zjemnění)
  - druhé vlákno asynchronně **udržuje seznamy**, přepočítává metriky (nařizuje provádění LoD operací)
  - další asynchronní vlákno může ulehčit vykreslování a provádět **LoD operace**

- **zpětná vazba** z vykreslovacího enginu: dolaďování konstant metrik a celkového (maximálního) počtu trojúhelníků

Spojité LoD

- **Progressive meshes** (Hugues Hoppe, 1996)
  - počáteční trojúhelníková síť + posloupnost elementárních operací
  - používá dříve publikované optimalizační techniky
  - hodí se i na postupný přenos dat (Internet) nebo kompresi
- **View-dependent PM** (Hugues Hoppe, 1997)
  - možnost nezávislého adaptivního zjemňování sítě
  - není již založena na lineárním předem připraveném postupu

Progressive Meshes I



- elementární operace s hranou:

Progressive Meshes II

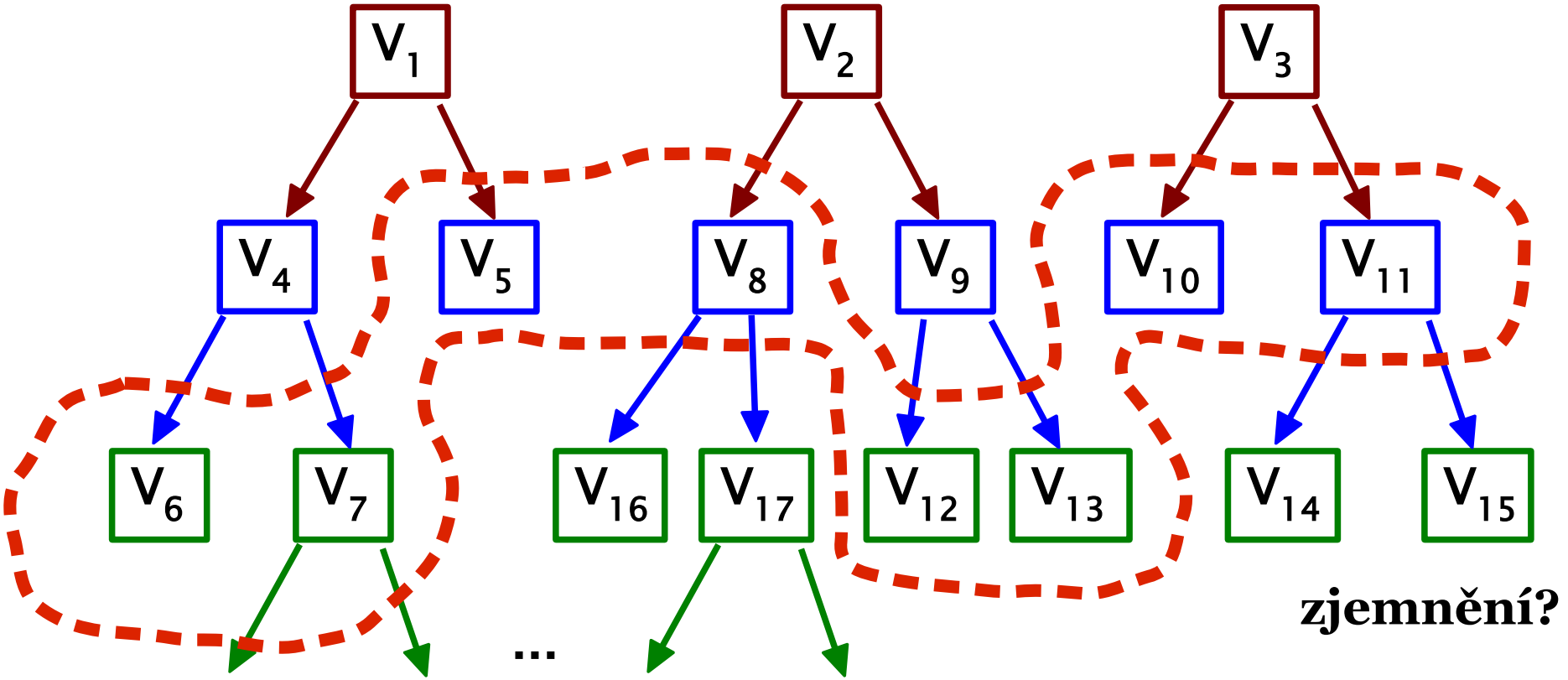
- **reprezentace** trojúhelníkové sítě
  - počáteční síť $\mathbf{M_0}$
  - posloupnost operací „vertex split“ $\mathbf{vs_0}$, $\mathbf{vs_1}$, ..
  - každá operace si pamatuje: [$\mathbf{V_s}$, $\mathbf{V_l}$, $\mathbf{V_r}$, $\mathbf{A}$]
    (tři odkazy na staré vrcholy * a atributy: poloha nových vrcholů)
- **geomorphs**
  - možnost postupné animace (jeden **vs**)
  - nové dva vrcholy se postupně „odpojují“ od otce

Progressive Meshes III

- **selektivní zjemňování**
  - podle potřeby se dá počáteční síť zjemňovat jinou posloupností $\mathbf{vs_{io}}$, $\mathbf{vs_{i1}}$, ..
  - operace, která „není na řadě“ se může provést, když tři vrcholy $\mathbf{V_s}$, $\mathbf{V_l}$, $\mathbf{V_r}$ již v síti existují
  - pořadí zjemňování určuje metrika
    - např. přítomnost vrcholu v zobrazovaném frustu nebo v obrysovém polygonu
  - kritické místo nesplňuje nutnou podmínku ⇒ indukované zjemňování
    - potenciálně velký graf závislostí

View-dependent Refinement of PM I

- selektivní systematické zjemňování založené na PM

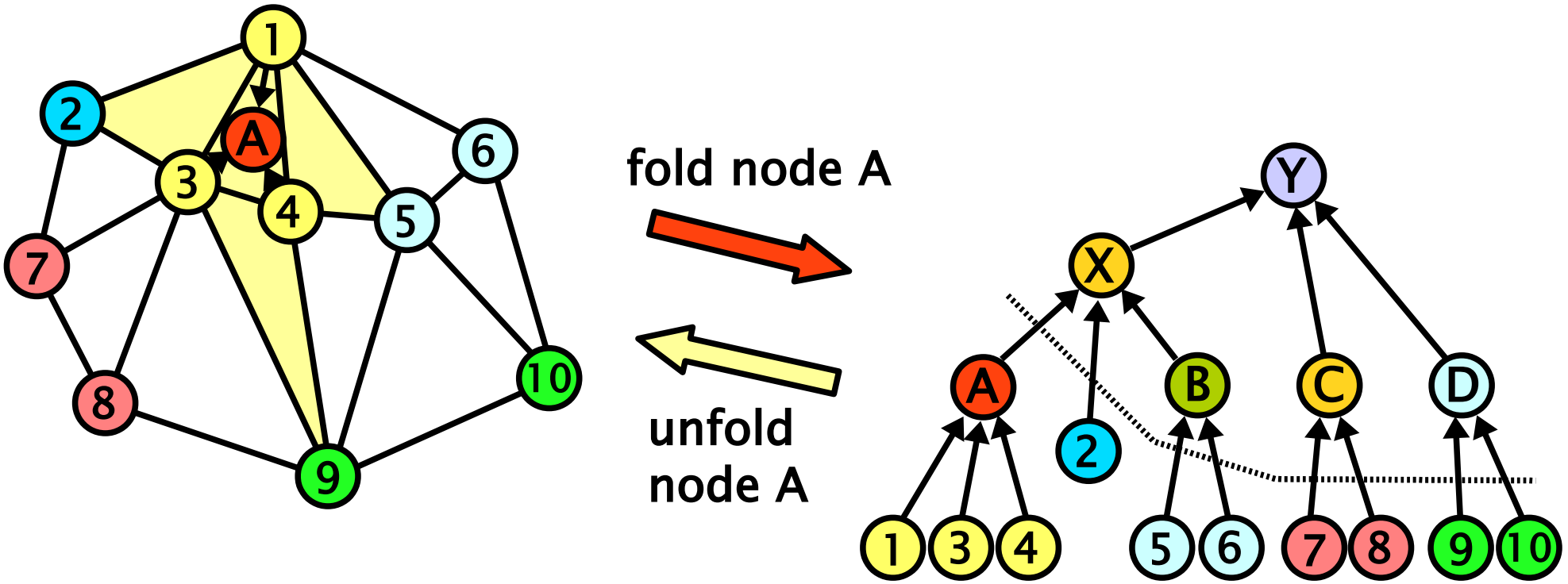


View-dependent Refinement of PM II

- **aktuálně zobrazená síť** je řezem v grafu („aktivní vrcholy“)
  - uzly grafu = vrcholy sítě
  - hrany = operace „vsplit“ (opak .. „edge collapse“)
- lokální zjemnění kresby .. řez se posune dolů,
  lokální zjednodušení .. řez se posune nahoru
- **realtime vykreslování**
  - cyklické procházení aktivních vrcholů
  - .. vyhodnocování, kde je potřeba síť zjemnit (a naopak)
  - amortizace: úprava sítě je asynchronní (zvláštní thread)

Hierarchie vrcholů

- předem připravená hierarchie vrcholů
  - používá koncept „proxy vrcholu“



Literatura

- C. Hansen, C. Johnson: *The Visualization Handbook*, Elsevier 2005, pp. 359-410
- D. Luebke, M. Reddy, J. D. Cohen, A. Varshney, B. Watson, R. Huebner: *Level of Detail for 3D Graphics*, Morgan Kaufmann, 2002, ISBN: 0321194969
- **http://research.microsoft.com/~hoppe/** (Hugues Hoppe – LoD metody)