Hlasovací zařízení řešící koncenzus v distribuovaném systému REPORT

Petr Kucera
kucerp28@fel.cvut.cz

26. ledna 2025

1 Úvod

Cílem projektu bylo zrobustnit algoritmus, který obsluhuje distribuovanou síť tvořenou hlasovacími zařízeními. Konkrétně bylo cílem ošetřit chybové stavy a navrhnout a implementovat přístup, který zajistí, že v síti bude moci být připojeno 1–N zařízení.

Síť je postavena na zařízeních ESP32, která jsou propojena pomocí protokolu ESP-NOW. Jedná se o speciální protokol, který spojuje horních pět vrstev OSI modelu do jedné, zvané ESP-NOW. Protokol má určité limity, ty také bylo nutné zohlednit při navrhování a implementaci. Konkrétně se jednalo o limit počtu sousedů a velikosti zprávy. Protokol definuje, že je schopen mít do 20 nešifrovaných sousedů a velikost zprávy může být maximálně 250 B.

2 Algoritmus a implementace

Algoritmus tvoří tři základní komponenty: registrace zařízení do DS, běžného chodu a terminace zařízení registrovaného v DS a plní základní 3 úkoly: synchronizaci času - zajišťuje *lídr* a udržuje si tak autoritu, distribuci logů - zajišťuje modifikovaný raft a distribuci seznamu zařízení distribuované sítě (DS) - zajišťuje modifikovaný raft.

2.1 Registrace zařízení

Jakmile se zařízení spustí, pošle *broadcast* zprávu typu $HELLO\_DS$. Zařízení v síti, která zprávu přijmou, si ho přidají na svůj seznam sousedů a zpět odešlou zprávou typu $NEIGHBOURS$. Do ní také přiloží základní informace o DS, jako je *ID epochy*. Jakmile zařízení obdrží více jak polovinu odpovědí. Přechází do další fáze – běžného chodu DS.

2.2 Běžný chod

Běžný chod se skládá z *epoch*. Každá epocha má svoje ID a tvoří ji dvě fáze – *volba lídra* a *běžný provoz*.

Zařízení bude označovat stejně jako v algoritmu Raft.

- *Lídr* ($MASTER$) funguje jako centrální prvek, který rozesílá logy událostí DS, seznamy sousedů a synchronizuje čas.
- *Následovník* ($SLAVE$) je pasivní zařízení, které se chová běžným způsobem.
- *Kandidát* je přechodná role v průběhu volby lídra.

První fází každé epochy je vždy **volba lídra**. Volba lídra se koná v každé epizodě, ovšem nemusí skončit úspěšně. Probíhá velice podobně jako volba lídra v algoritmu Raft. *Lídr* si udržuje svoji autoritu pomocí rozesílání zprávy pro synchronizaci času $TIME$. Pokud zařízení neobdrží zprávu do *timeoutu* t_{sync}, zařízení zvýší číslo epochy a spustí nové volby. Ty probíhají tak, že rozešle na všechny známé aktivní sousedy zprávu $REQUEST\_VOTE$, tedy žádost o to, že chce být *lídrem*. Poté může nastat jedna ze těchto tří situací:

- zařízení dostane **potvrzení od většiny** $GIVE\_VOTE$ aktivních sousedů a stane se novým *lídrem*
- nebo **přijme zprávu synchronizující čas** $TIME$, novým *lídrem* se stalo nějaké zařízení rychleji
- nebo budou **volby neúspěšné** do *timeoutu* $t_{election}$, volby skončí neúspěchem a začne nová epocha.

Po úspěšných volbách probíhá fáze **běžného provozu**. Pokud nejsou volby úspěšné, nemusí k ní dojít. Během ní DS distribuuje logy a zprávy typu *LOG*. Při registraci zařízení do DS nebo terminaci rozesílá seznam sousedů. A lídr posílá synchronizační zprávy $TIME$ pro udržení autority *lídra* a synchronizaci času v DS.

Pokud v této aplikaci dojde k selhání zařízení, neexistuje žádná opravná rutina, a to z podstaty systému a navrženého algoritmu. Respektive neexistuje stav, který by algoritmus neřešil a šlo ho softwarově opravit. Pokud dojde k selhání zařízení, uživatel bude informován pomocí stavu zařízení.

Chyba ožití starého *lídra* je ošetřena mechanismem *ID* epochy. Zařízení akceptuje pouze zprávy s aktuálním nebo vyšším *ID* epochy.

2.3 Terminace zařízení

Jako **neaktivní zařízení** je označeno každé, které nepřijme více jak tři zprávy úspěšně. Pokud neodpovídá, je terminováno z DS. Respektive je v seznamu sousedů označeno jako neaktivní a tato informace je pak rozeslána přes *lídra* do celého DS.

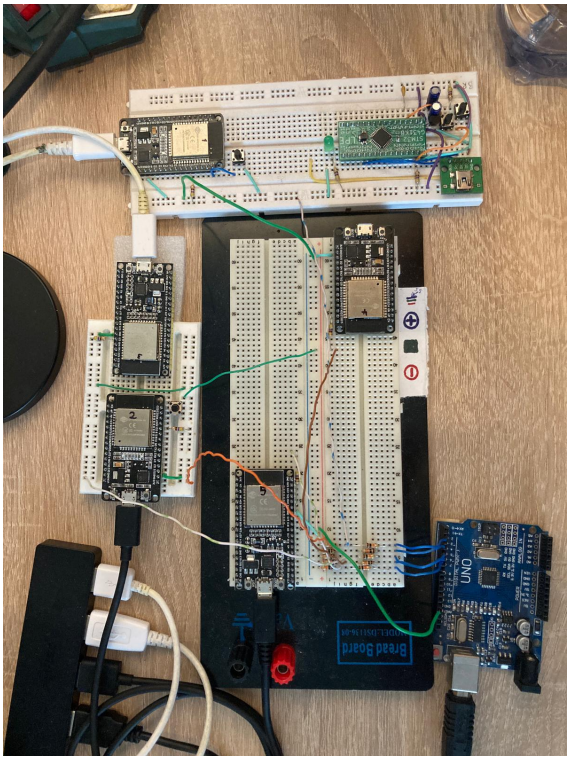
2.4 Přizpůsobení dynamickému počtu sousedů

Implementace mění statický počet sousedů na počet dynamický a umožňuje tak zapojit do sítě až 20 zařízení.

Při určité konfiguraci by určitě bylo možné vymyslet alternativní řešení pro využití více sousedů, např. vytvořit více subsítí. Pro tuto aplikaci to ovšem nedává smysl, proto jsem se rozhodl respektovat tento limit a zohlednit ho při následném výpočtu paměťové náročnosti.

V algoritmu jsem pozměnili především následující části.

- **Ukládání**: V podstatě jsou 2 možnosti pro ukládání dat do paměti - statická nebo dynamická cesta. Statická by znamenalo mít neustále alokováno 20 x 16 bytes, kde by se do pole zapisovala aktuální zařízení. Druhým způsobem je paměť dynamická - např. linked list. Limitací technologie ESP-NOW jsem zvolil první možnost, tedy statický styl ukládání a to z důvodu bezpečnější implementace (resp. méně náchylné na chybu) a z důvodu menší výpočetní zátěže pro práci s paměti.
- **Změna struktury packetu**: Součástí každé zprávy momenátlně byl i seznam sousedů. To nyní změním a to z důvodu omezení paměti. Místo části framu, kde doteď byli uložení sousedé (neighbor), uložím jednoduchý checksum, který bude typu $uint32\_t$ a bude preprezentovat sousedy následujícím způsobem. Za každý node ve stavu, $NOT\_INITIALIZED$ bude přičtena hodnota 1, $INACTIVE$ bude přičtena hodnota 100, $ACTIVE$ bude přičtena hodnota 10000.
- **Nový typ zprávy**: Vytvářím nový typ zprávy, který bude odesílát seznam sousedů. Jelikož je velikost zprávy maximálně 250 bytes, budu muset seznam rozdělit do dvou zpráv. To ale ničemu nevadí. Tato zpráva se bude odesílat vždy, po zachycení zprávy 'HELLO_DS' a detekci naktivního zařízení.



Obrázek 1: Ukázka testování provozu

3 Spuštění kódu

Pro spuštění kódu je třeba mít nainstalovaný *framework esp-idf*.

```
clone https://github.com/espressif/esp-idf.git # prepnete na vhodny release,
    testovano pro v5.4
cd esp-idf
./install.bat # ci jiny vhodny installer pro pouziany system
```

Pro kompilaci je vhodné využít nástrojů poskytovaných frameworkem.

```
idf.py reconfigure # stazeni a instalace komponent esp-now
idf.py build
idf.py flash
idf.py monitor
```

Doporučuji nastavit automatické spuštění exportu. Pokud využijete již předpřipraveného vscode envirometal a změníte cestu k esp-idf knihovně, provede se export vždy automaticky při spuštění terminálu.

Kód je také dostupný v repositáři https://github.com/petrkucerak/rafting-button/tree/main/code/dynamic-neighbors.